AN10207

Smart Card reader application with TDA8029 Mask 06 and Mask 07

Rev. 1.0 — 2 February 2011 Application note

Document information

Info	Content
Keywords	TDA8029, Smart card interface, ISO 7816-3 and ISO 7816-4, E.M.V. 4.0
Abstract	This application note describes the software implemented in both TDA8029 mask 06 (TDA8029HL/C206) and TDA8029 mask 07 (TDA8029HL/C207) in order to handle a communication between a system controller and a smart card. Both mask 06 and mask 07 can support all the asynchronous smart cards using either T=0 or T=1 protocol and some synchronous smart cards (S=9, S=10 and I2C). Mask 07 fixes an issue in the case 4 APDU command with" Le" parameter less or equal to 2 (see annex IV). The control of the TDA8029 by the host controller is done using a RS232 serial interface.

Revision history

Rev	Date	Description
1.0	20110202	Initial release

Contact information

For more information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

1. Introduction

TDA8029 is a smart card coupler providing all the analogue electrical interface signals to the smart card. This coupler is able to manage asynchronous cards due to its specific ISO7816 UART and to its embedded 80C51 microcontroller core; it can also manage synchronous cards such as I2C cards or prepaid telephone cards.

The software embedded in this device is able to support any ISO 7816 asynchronous smart card (T=0 or T=1 protocol) and some synchronous cards (I2C, S9 and S10).

It completely handles the communication layer between the card and the host system.

A specific protocol called “ALPAR” has been defined on the serial interface between TDA8029 and the host system; it uses the APDUs frame types to convey the asynchronous card commands and specific frames for the synchronous cards. A dedicated command has been added to carry TPDUs frames for T=1 protocol only.

A board has been built in order to demonstrate a communication between a smart card and a host system.

When the host is a PC communicating with the TDA8029 using a RS232 link, a software called SCRTester can be used (see detailed description in Annex I, page 46).

The following diagram illustrates this application.

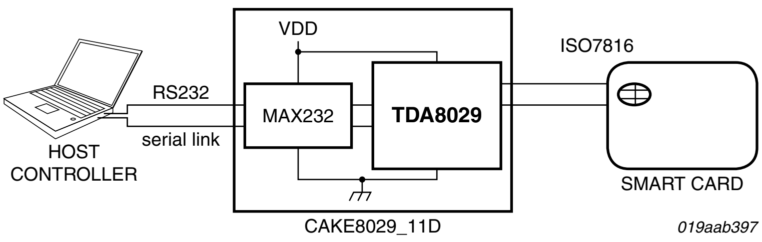


Fig 1. TDA8029 application diagram example

2. Hardware and power management

2.1 Hardware

The board CAKE 8029_11D (see Annex II, page 48) has been made to demonstrate the features of the TDA8029 with both Mask 06 and Mask 07 software.

This board is supplied under +2.7 V to +6 V.

In case of RSR232 host interface configuration, it may be connected to a PC by means of the serial port.

Depending on the PC which is used, the communication between the PC and the board can be fixed at different baud rates (from 4800 to 115200 baud).

The default baud rate is 38400 and it can be changed by a special command in SCRTester (set_serial_baud_rate on page 29).

In case of I2C-bus host interface configuration, it should be connected to the I2C-bus master using *SDA* and *SCL* lines and possibly *WakeUpSlave* and *SlaveI2CMute* lines.

2.2 Host Controller Interface

The TDA8029 with mask 06 or mask 07 software can be interfaced to the host controller by using either a RS232 serial link or an I2C-bus.

The choice of the interface is done according to the state of P17 (i.e. pin 1 of the TDA8029) at the powering on or at the reset of the TDA8029.

Table 1. Host interface hardware configuration

Interface	P17 (#1)
RS232 (see section 5)	**VDD**
I2C (see section 6)	**GND**

2.3 Power management: Energy Saving Mode

In order to benefit from the low power features of the TDA8029, both Mask 06 and mask 07 software implements a special management of the TDA8029 called Energy Saving Mode (ESM).

In this mode, outside an exchange of commands between the host and the TDA8029, the card clock is either switched off (level high or low) or set to Fint/2 depending on the clock stop mode described in its ATR and the microcontroller is set in power down mode. This mechanism allows a lower average current consumption for the TDA8029.

This mode cannot be used for EMV approval.

A general description of ESM mechanism can be summarized on the following figure:

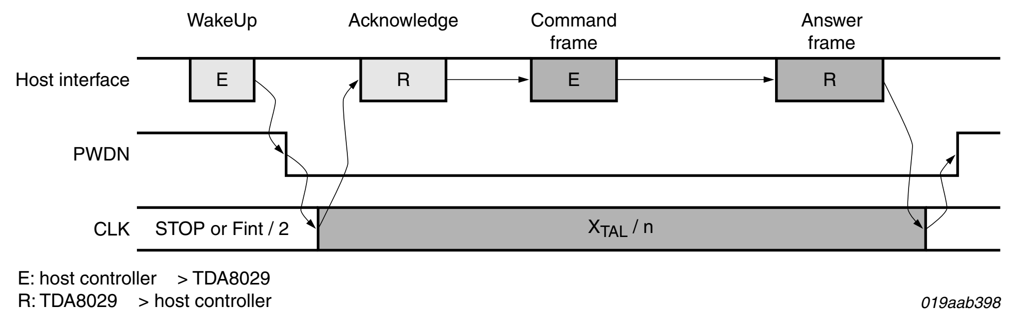


Fig 2. ESM Mechanism

In this figure, the “host interface” symbolizes the link between the host controller and the TDA8029: it may be either a RS232 link or an I2C-bus.

When the host wants to send a command frame to the TDA8029, it first wakes it up. The waking up process is different according to the type of the interface used. It will be detailed hereafter (see section Dialog structure in case of Energy Saving Mode activated§5.2 and §6.3).

As soon as it is able to receive and treat a command frame, the TDA8029 indicates it to the host controller using an acknowledge process. When using an I2C-bus interface, this stage may be possibly skipped (as the I2C-bus mechanism already implements an acknowledge notion).

Then, the host controller can send the complete command frame to be treated by the TDA8029.

When the TDA8029 has finished the execution of the command, it sends back the corresponding answer to the host.

When the exchange is completed, the TDA8029 goes back in power down mode after having switched the clock of the card to the lower current consumption available configuration. If the card does not support the clock stop mode or does not specify it in its ATR, the clock will be set to Fint/2. Note that a command (set_esm_properties, page 29) allows to force a clock stop mode even if the card does not specify it. Stop the clock rather than leave it at Fint/2 provides a significant power consumption saving.

To activate this mode P26 (i.e. pin 25 of the TDA8029) has to be connected to V_{DD} or to be left open at the powering on of the board (or at the reset), whereas connecting P26 (pin 25) to ground will force the TDA8029 to never enter into this energy saving mode.

Table 2. ESM hardware configuration

ESM	P26 (#25)
ON	VDD or not connected
OFF	GND

2.4 Shutdown mode

The TDA8029 can be set to shutdown mode using the shutdown pin SWDN (pin #5). When this pin is switched to low state then the bit SDWN within HSR will be set, causing an interrupt on INT0 pin (#29).

The TDA8029 will read the status, deactivate the card if it is active, set all ports to 1 and enter in power down mode by setting the bit PD in the C51 PCON register. In this mode, it will consume less than 20 µA.

When the shutdown pin is set to high state back, a power-on-reset operation is performed so the chip recovers the same state than at power on.

The host controller should observe a waiting time before sending commands to the TDA8029. This waiting time depends on the value of capacitor connected to the pin CDEL (#6), determining the power on reset pulse width. Typically, this pulse width is 1 ms par 2 nF (see next chapter).

2.5 Power-On Reset

The CDEL pin is used to add a delay between a power on or a supply dropout, and the effective start of the chip. This delay is used as a reset pulse for the internal controller.

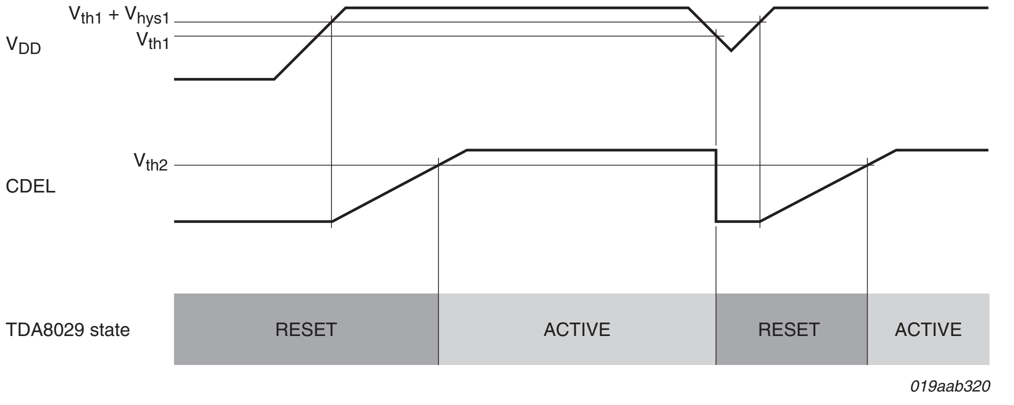


Fig 3. CDEL pin action overview

The delay is defined by the load time of the capacitor connected on the pin: 1 ms per 2 nF typically as shown in figure 4, next page.

The minimum advised value for the capacitor on CDEL is 22 nF.

In some application, a bigger delay can be needed when the overall system reset needs more than 10 ms (22 nF typical capacitor value). In this case the below figure allows to select the right capacitor value.

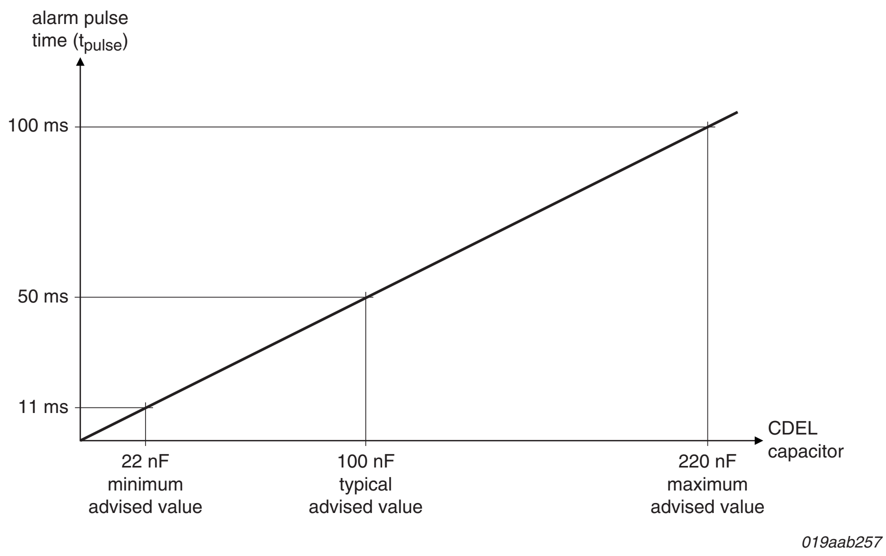


Fig 4. Correspondence between reset delay and CDEL capacitor

3. Software aspect

Both mask 06 and mask 07 has been developed in order to be used either in ISO7816-3 or E.M.V. 4.0 environment. It is compliant with erratum #13 Test Bench Description – Executable Tests TBD/EXE/T01 3.0, November 21st 2002.

Some specific error messages are dedicated to the E.M.V. environment (ATR parameters not allowed).

4. Protocol “ALPAR”

The communication between the host controller and the TDA8029 obeys to a protocol named ALPAR.

This protocol encapsulates the useful data of a message in an invariant frame structure and defines a dialog structure of messages exchanges.

Data is exchanged between the host controller and TDA8029 in blocks, each made up of binary characters on one byte:

4 header characters

0 to 506 data characters (C-APDU or R-APDU)

1 LRC character

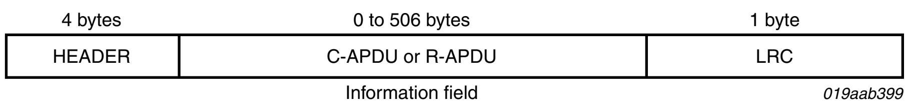


Fig 5. Frame structure

The 4 header bytes include the following bytes:

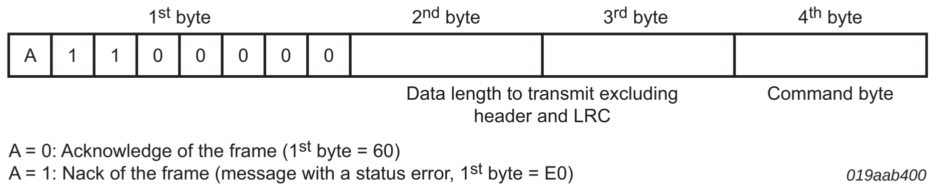


Fig 6. 4 header bytes

The LRC (Longitudinal Redundancy Check) byte is such that the exclusive-oring of all bytes including LRC is null.

4.1 General dialog structure

The host controller is the master for the transmission; each command from the master is followed by an answer from TDA8029 including the same command byte as the input command.

However, in some cases (card insertion or extraction, a time out detection on Rx line or an automatic emergency deactivation of the card) the TDA8029 is able to initiate an exchange.

4.1.1 Successful command

System to TDA8029:

60	XX XX	YY	nnnnnnnnnnnnnnnnnn	ZZ
ACK	length	code	Data (C-APDU)	LRC

TDA8029 to System:

60	UU UU	YY(1)	mmmmmmmmmmmmmm	TT
ACK	length	code	Data (R-APDU)	LRC

019aab401

(1) The same command byte YY is returned in the answer from TDA8029.

Fig 7. Successful command frame description

4.1.2 Unsuccessful command

System to TDA8029:

60	XX XX	YY	nnnnnnnnnnnnnnnnnn	ZZ
ACK	length	code	Data (C-APDU)	LRC

TDA8029 to System:

E0	UU UU	YY	SS(1)	TT
NACK	length	code	Status	LRC

019aab402

(1) In that case, the status contains the error code information (see error list)

Fig 8. Unsuccessful command frames

4.1.3 Answer with an acknowledge (power_off, idle_mode, power_down_mode)

System to TDA8029 (example: power_off):

60	00 00	4D	2D
ACK	length	code	LRC

TDA8029 to System:

60	00 00	4D	2D
ACK	length	code	LRC

019aab403

(1) In the case where the answer is an acknowledge of the command, the TDA8029 sends back a frame with the same content of the command.

Fig 9. Acknowledge frame

5. RS232 Interface

5.1 General description

The serial interface between the TDA8029 and the host controller is a full duplex interface using the two lines RX and TX.

RX (pin 32) is used to receive data from the host controller; TX (pin 31) is used to send data to the host controller.

No flow control or supplementary line is used (no hand check).

The serial data format used is:

- start bit
- data bits
- stop bit, no parity

The default baud rate is 38400 baud, but it can be changed (from 4800 to 115200 baud) by a host command set_serial_baud_rate (page **29**).

Warning: in order to comply with standard baudrates values (4800, 9600, 19200…), the crystal used with TDA8029 must be fixed to the 14.745 MHz as defined in the reference schematics. There is no big constraint on its precision (must be less than 1 %).

If the crystal frequency is different, then the baudrate is changed accordingly.

e.g. If the crystal freq. is equal to 10 MHz, then the default baudrate will become:

10 x 38400 / 14.745 = 26042 bps.

In case of RS232 interface mode configuration, the TDA8029 has to be connected as follows:

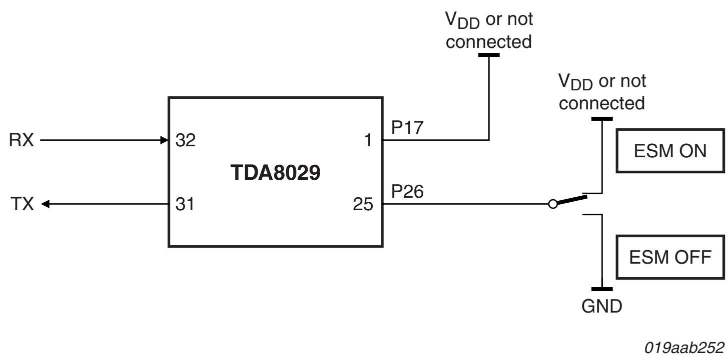


Fig 10. RS232 Interface configuration

5.2 Dialog structure in case of Energy Saving Mode activated

As it is explained in §2.3, both Mask 06 and mask 07 implement a special management of the TDA8029 for energy savings purpose. This Energy Saving Mode is activated when P26 (i.e. pin 25) is tied to V_{DD} of left open at reset of the TDA8029.

Due to this Energy Saving Mode implementation, the serial interface is adapted as follows.

5.2.1 Communication initiated by the system controller

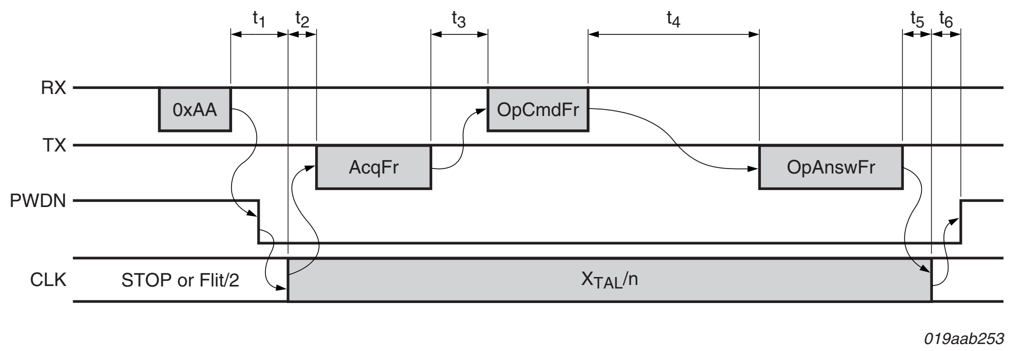


Fig 11. Communication initiated by the system controller

To initiate a normal exchange with the TDA8029, the host has to first send a specific frame composed of only one character (0xAA) to wake up the TDA8029. When the TDA8029 is completely waked up, it sends an acknowledged frame back to the host.

System to TDA8029 (Specific waking up frame)

AA

TDA8029 to System (Acknowledgement frame)

60	00 00	BB	DB
ACK	length	code	LRC

After this synchronization stage, the operational communication can start as described in 4.1.

System to TDA8029:

60	XX XX	YY	nnnnnnnnnnnnnnnnnnn	ZZ
ACK	length	code	Data (C-APDU)	LRC

TDA8029 to System:

60 or E0	XX XX	YY	nnnnnnnnnnnnnnnnnnn	ZZ
ACK or NACK	length	code	Status or Data (R-APDU)	LRC

After having sent its complete frame, the TDA8029 returns in power down.

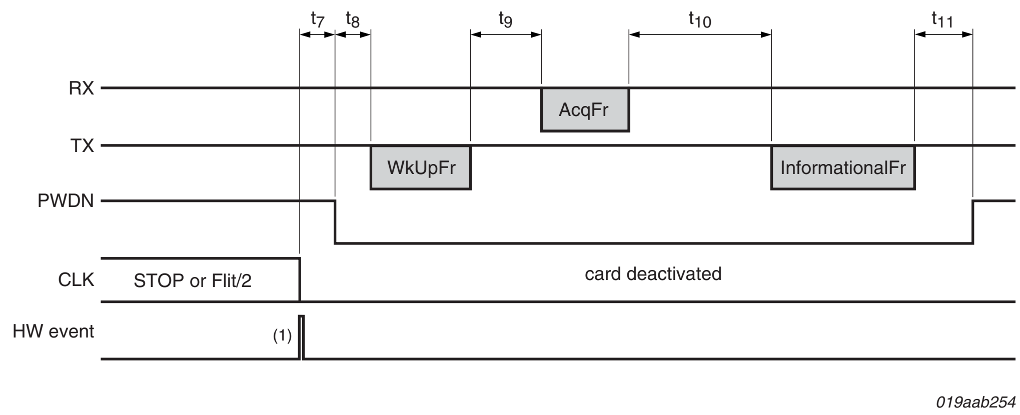
019aab404

Fig 12. Communication initiated by the system controller

5.2.2 Communication initiated by the TDA8029

Any card event (extraction or insertion, overcurrent on VCC or RST, overheating) will wake up the TDA8029.

In that case, this is the TDA8029 which initiates the communication.



(1) In this example the card is deactivated due to hardware event

Fig 13. Communication initiated by the TDA8029

It first sends the following specific frame to warn the host that it wants to send data. This frame is the same than the one used to acknowledge a received waking-up character (section 5.2.1) except that the length in that case is equal to one (as it is equal to zero in case of a normal acknowledge frame).

TDA8029 to System

60	00 01	BB	01	DB
ACK	length	code	data	LRC

After that, the TDA8029 waits until the host replicates this frame.

System to TDA8029

60	00 01	BB	01	DB
ACK	length	code	data	LRC

Then, the TDA8029 sends the informational frame and returns in power down mode.

TDA8029 to System:

60 or E0	XX XX	YY	nnnnnnn	ZZ
ACK or NACK	length	code	data	LRC

019aab405

Fig 14. Communication initiated by the TDA8029 – Frame description

For instance, a card insertion will generate the following informational frame:

TDA8029 to System

60	00 01	A0	01	C0
ACK	length	code	data	LRC

Whereas in the case of a card extraction, the frame will be:

TDA8029 to System

60	00 01	A0	00	C1
ACK	length	code	data	LRC

019aab406

Fig 15. Card move – frame example

5.2.3 Timing considerations

The following timings are referenced in the figures page 11 and 12.

- t_1: Waking up and clock switching time (typically 550 µs).
- t_2: TDA8029 reaction time (typically 33 µs).
- t_3: Host-dependant reaction time. (no limit).
- t_4: Process time (depends on the type of the command frame).
- t_5: Clock switching time (typically 80 µs).
- t_6: Power Down setting time (typically 0.3 ms).
- t_7: Waking up and clock switching time (typically 550 µs).
- t_8: TDA8029 reaction time (typically 630 µs).
- t_9: Host-dependant reaction time. (no limit).
- t_{10} : TDA8029 reaction time (typically 630 µs).
- t_{11} : Power Down setting time (typically 0.3 ms).

5.2.4 Abnormal communication process

Three different cases can be met:

- Time out detected on Rx line (more than 10 ms between the leading edge of two characters inside the command frame sent by the host controller). As soon as the time out is detected, the TDA8029 sends back an error frame:

System to TDA8029	0xAA
TDA8029 to system	0x60 0x00 0x00 0xBB 0xDB
System to TDA8029	ACK Length Code (Parameters) LRC (*erroneous frame*)
TDA8029 to system	NACK 0x00 0x01 Code 0xFF LRC

Then the TDA8029 goes back to power down mode.

- Card extraction detected during a card IO card session: if the TDA8029 detects a card extraction as it is processing an APDU with the card, it returns two consecutive messages back to the host controller

System to TDA8029	0xAA
TDA8029 to system	0x60 0x00 0x00 0xBB 0xDB
System to TDA8029	ACK Length 0x00 Parameters LRC (*C-APDU*)
TDA8029 to system	NACK 0x00 0x01 0x00 0xC0 LRC (*card absent*)
TDA8029 to system	ACK 0x00 0x01 0xA0 0x00 LRC (*card extraction*)

Then the TDA8029 goes back to power down mode.

- Unexpected reception detected during a communication process; the TDA has not finished to process a received command frame (it has not sent completely its answer frame while the host controller sends a new command frame): in that case, the TDA8029 sends the correct answer to the first received command and then sends the error frame informing the host that it has lost at least a command frame.

System to TDA8029	0xAA
TDA8029 to system	0x60 0x00 0x00 0xBB 0xDB
System to TDA8029	ACK Length Code1 (Parameters) LRC
System to TDA8029	ACK Length Code2 (Parameters) LRC
TDA8029 to system	ACK or NACK Length Code1 (Parameters) LRC
TDA8029 to system	NACK 0x00 0x01 Code2 0xF1 LRC

Then the TDA8029 goes back to power down mode.

6. I2C-bus interface

6.1 General description

As specified in the I2C-bus specification, only two lines may be used to manage the serial link between the TDA8029 and the system controller:

- a serial data line (SDA), has to be connected to RX (pin 32 of the TDA8029)
- and a serial clock line (SCL), has to be connected to P16 (pin 2 of the TDA8029).

In addition to I2C specification, two other lines can be used to manage Energy Saving Mode mechanism:

- **WakeUpSlave**,line used to wake up the TDA8029 before sending an I2C frame to it has to be connected to INT1 (pin 30 of the TDA8029)
- **SlaveI2CMute**,line used by the TDA8029 to indicate to the host controller either that it is ready to receive a command frame or to send the corresponding answer, or to signal an hardware event has to be connected to P27 (pin 24 of the TDA8029).

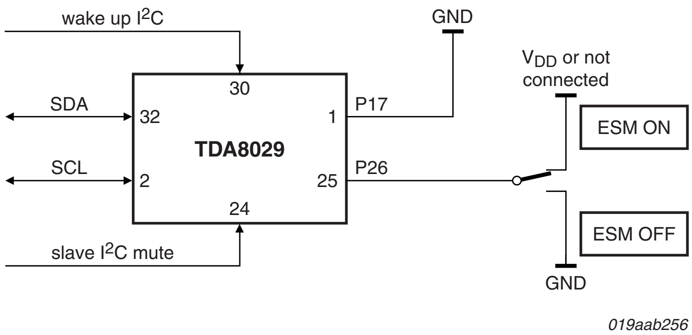


Fig 16. I2C Bus interface configuration

In fact, ESM may be used even with a pure 2 lines I2C-bus.

In that case, the pin 30 of the TDA8029 has to be connected together with SDA line.

Table 3. I2C modes

I2C Mode	ESM	RX (#32)	P26 (#26)	INT1 (#30)	P27 (#24)
2 lines	**OFF**	SDA	GND	nu	nu
2 lines	**ON**	SDA	VDD	SDA	nu
4 lines	**ON**	SDA	VDD	WakeUpSlave	SlaveI2CMute

As the system controller is the I2C-bus master, it will initiate all the exchanges. Each command from the master is followed by an answer from TDA8029.

Normally, as it is a I2C-bus slave, the TDA8029 can not warn the host controller by means of the bus when an hardware event happens (abnormal deactivation of the card, movement detection, …). The line SlaveI2CMute can be used for that.

When such an event occurs, the TDA8029 falls down the SlaveI2CMute line so that the host controller can known that something has happened on the reader side.

The host controller can send a GetReaderStatus (see section 9.1.3) command frame to receive details on the current state of the TDA8029.

This particularity may be deactivated by means of the set_esm_properties command (see page 29). In that case, the TDA8029 does not inform the host controller when a specific event happens on the card. Of course, the TDA8029 takes in charge the security of the card and automatically deactivates it if needed.

6.2 Energy Saving Mode deactivated

When the TDA8029 is not configured in Energy Saving Mode, it is able to accept a command sent by the host controller as soon as it has finished to handle the previous one (no waking up delay).

On the other hand, after having received a complete command frame from the host, the TDA8029 will need a variable delay time to achieve the related task before to send back an answer to the host.

This time depends on the kind of the command, the kind of the card (baudrate, CWT, BWT, …), the length of the card exchange, and so on…

During this execution time, the TDA8029 will be mute, i.-e. it will not acknowledge any incoming messages from the host controller.

Once the command message is processed, the TDA8029 will be available to give its answer when the host controller will address it.

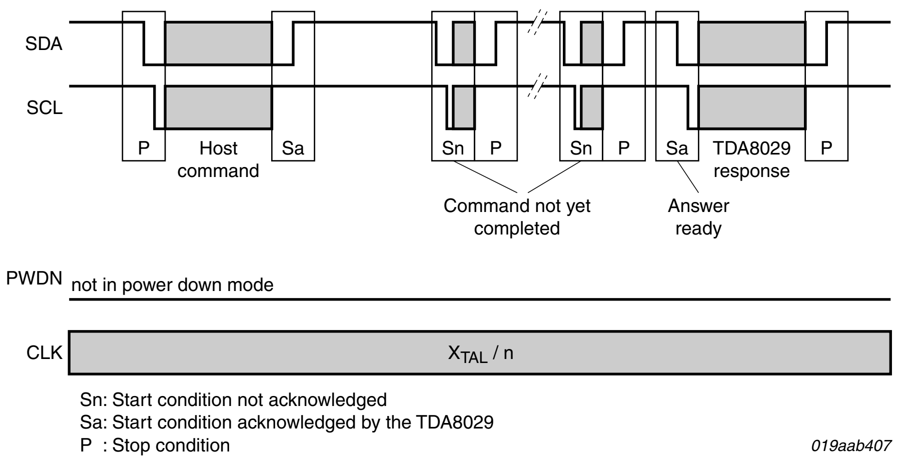


Fig 17. I2C communication with ESM off

6.3 Energy Saving Mode activated

When used in Energy Saving Mode, the goal is that the TDA8029 stays in power down mode outside an I2C-bus exchange with it.

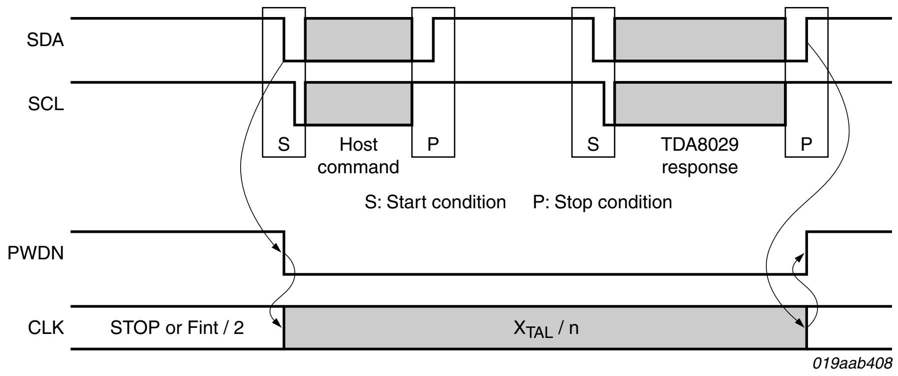


Fig 18. I2C communication with ESM on

6.3.1 4 lines I2C-bus

If the host controller sends directly an I2C frame to the TDA8029 as it was asleep, it will not be able to acknowledge its address (due to waking-up delay).

Consequently, the lines WakeUpSlave and SlaveI2CMute are used to manage the complete exchange:

1 - Before sending a frame to the TDA8029, the host controller wakes it up with a negative pulse on WakeUpSlave line (minimum duration of 0.8 µs)

2 - As soon as it is completely waked up, the SlaveI2CMute line falls down,

3 - The host controller can now send the I2C write command frame. The SlaveI2CMute line goes up after the TDA8029 has recognized its I2C address (0x50)

4 - Once the command frame is received, the TDA8029 processes it. When it is ready to give corresponding results to the host controller, the SlaveI2CMute line falls down again

5 - The host controller can now send I2C read command frame. The SlaveI2CMute line goes up after the TDA8029 has recognized its I2C address (0x51)

6 - Once the answer has been totally read by the host controller, the TDA8029 returns in power down mode until the next host controller exchange.

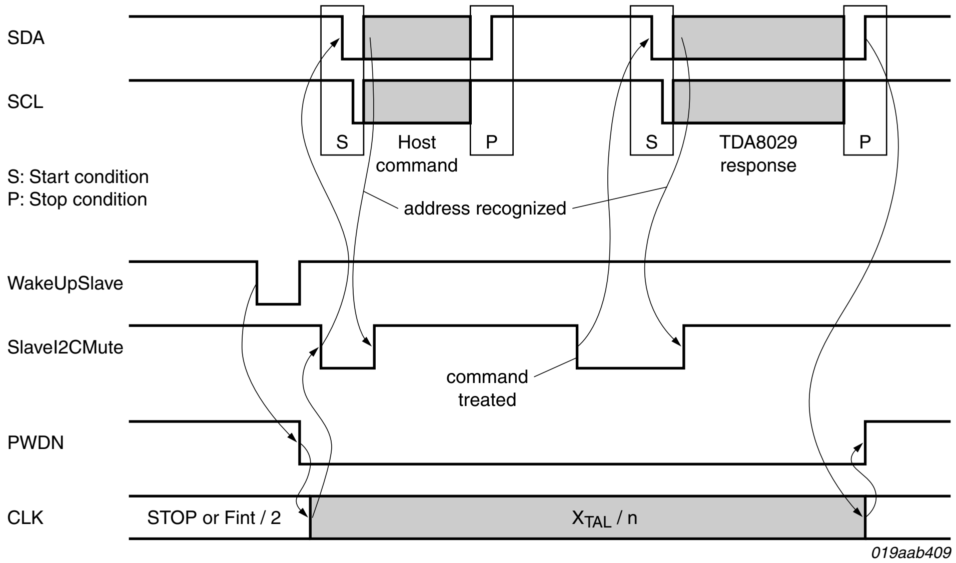


Fig 19. 4 lines I2C communication example initiated by the host

Moreover, in that configuration, if the TDA8029 has to warn the host controller to inform it about a hardware event, it can do it by falling down the SlaveI2CMute line outside a normal exchange. Then, the host sends a command with GetReaderStatus (see section 9.1.3) opcode to get detailed information from the TDA8029. In that case, the host controller should not use the WakeUpSlave line as the TDA8029 is already waked up.

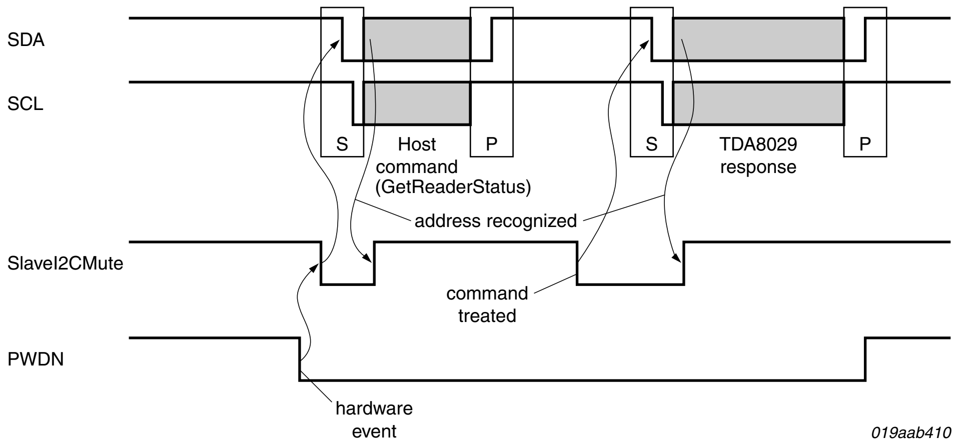


Fig 20. 4 lines I2C communication example initiated by the TDA8029

Remark: The main advantage of this mode (4 lines I2C-bus) is that the current consumption of the TDA8029 is completely optimized.

When the I2C-bus master addresses to other slave than the TDA8029, the latter is not waked up (this is not SDA or SCL line which wakes it up but the dedicated WakeUpSlave line).

6.3.2 2 lines I2C-bus

As described in the table in §6.1, the Energy Saving Mode can be used even with a host controller using a pure I2C-bus interface, without the two additional lines *WakeUpSlave* and *SlaveI2CMute*.

In that case, the SDA line has to be connected on pin 30 of the TDA8029 in addition to pin 32.

Thus, when the TDA8029 is asleep, every frame on the I2C-bus wakes it up; even if the frame is not addressed to the TDA8029. That is the main drawback (according to the current consumption) of this configuration.

As soon as it recognizes its address on the I2C-bus, the TDA8029 acknowledges it and then the normal exchange can go.

The embedded microcontroller needs a delay time before to come back completely operational when waked up by a I2C-bus frame. During this period, the I2C command will not be acked by the TDA8029 and the host has to try again until its command will be correctly acked.

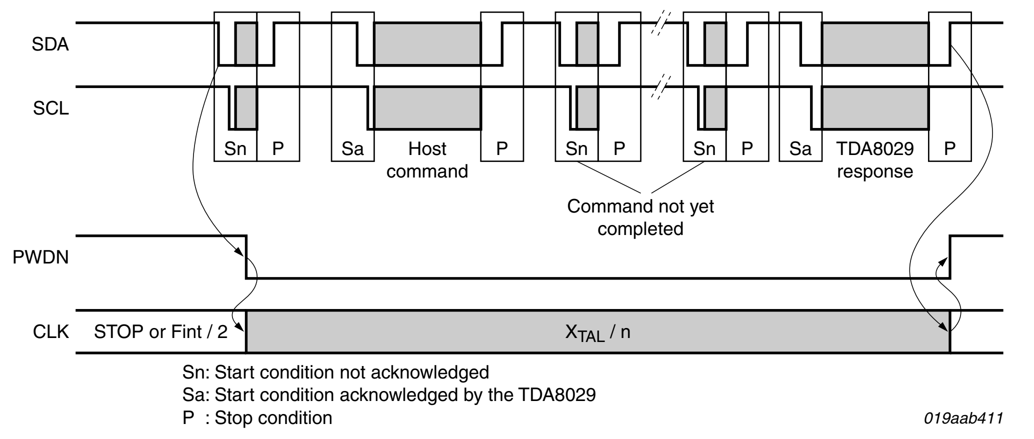


Fig 21. 2 lines I2C communication

6.4 Data link layer

I2C-bus slave:

Slave address: 0x50

Frequency:

Maximum SCL frequency: 60 kHz

For further details on restrictions on I2C bus, see Annex V: Recommendation rules when using the I2C interface page 63.

Clock synchronizing:

As an I2C-bus slave, the TDA8029 can slow down the bus clock by extending each clock low period. The speed of any I2C-bus master is thereby adapted to the internal operating rate of the TDA8029.

This synchronizing mechanism is also called clock stretching.

6.5 I2C transactions

The I2c transactions use the protocol ‘ALPAR’ described previously.

6.5.1 I2C write command message

The host system sends the following I2C message structure to the TDA8029:

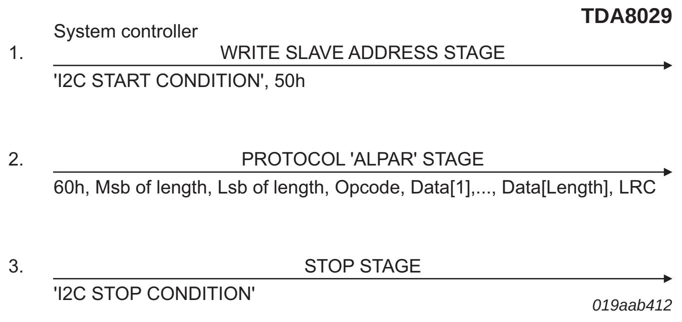
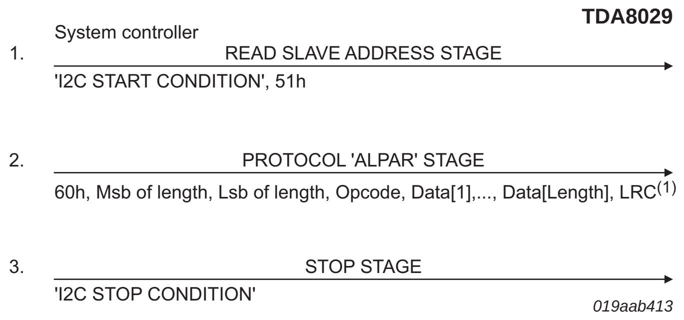


Fig 22. I2C write protocol

6.5.2 I2C read command message, normal answer

In this case, the I2C message structure is:

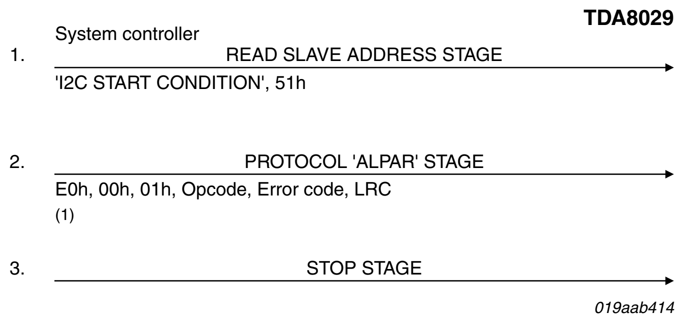


(1) The pattern 60h indicates a normal answer from the TDA8029

Fig 23. I2C read protocol

6.5.3 I2C read message, error answer.

The I2C message structure is shown below:



(1) The pattern E0h indicates an error answer from the TDA8029.

Fig 24. I2C read protocol - Error

6.5.4 Examples of I2C transactions

This table below shows some I2C transactions between the host system and the TDA8029. The data sent by the TDA8029 are in bold:

Table 4. I2C communication examples

Opcode	I2C write message	I2C read messages
Check card presence	S, 50, **60, 00, 00, 09, LRC**, P	If card present: S, 51, **60, 00, 01, 09, 01, LRC**, P If card absent: S, 51, **60, 00, 01, 09, 00, LRC**, P
Mask Number	S, 50, **60, 00, 00, 0A, LRC**, P	S, 51, **60, 00, 0E, 0A, data, LRC**, P
Power up card 3v, E.M.V.	S, 50, **60, 00, 01, 6D, 01, LRC**, P	If card powered up successfully: S, 51, **60, 00, 0F, 6D, ATR, LRC**, P If card absent, error message: S, 51, **E0, 00, 01, 6D, C0, LRC**, P
Power up card 5v, ISO	S, 50, **60, 00, 01, 6E, 00, LRC**, P	If card powered up successfully: S, 51, **60, 00, 0D, 6E, ATR, LRC**, P If card absent, error message: S, 51, **E0, 00, 01, 6E, C0, LRC**, P
Power off card	S, 50, **60, 00, 00, 4D, LRC**, P	S, 50, **60, 00, 00, 4D, LRC**, P

Note: In the I2C messages given above:

- ‘S’ stands for I2C start condition
- ‘P’ means I2C stop condition.

7. Command bytes

7.1 General commands

The following command bytes are available (listed in numerical order):

Table 5. Command summary

Command	Code	Answer from reader	page
card_command (APDU)	00_H	Card response (APDU) or error message	32
process_T=1_command	01_H	T=1 frame or error message	33
write_I2C	02_H	Acknowledge or error message	41
read_S9	03_H	Data read from the card or error message	37
read_S9_protection	04_H	Data read from the card or error message	37
write_S9_protected	05_H	Acknowledge or error message	37
write_S9_unprotected	06_H	Acknowledge or error message	38
verify_pin_S9	07_H	Acknowledge or error message	38
compare_S9	08_H	Acknowledge or error message	38
check_pres_card	09_H	Indication of the card presence	28
send_num_mask	$0A_H$	1 parameter giving the mask number	28
set_card_baud_rate	$0B_H$	Acknowledge	35
ifsd_request	$0C_H$	Acknowledge or error message	34
set_serial_baud_rate	$0D_H$	Acknowledge or error message	29
negotiate (PPS)	10_H	Acknowledge or error message	33
set_clock_card	11_H	Acknowledge or error message	34
read_I2C	12_H	Data read from the card or error message	40
read_2C_extended	13_H	Data read from the card or error message	40
read_current_I2C	23_H	Data read from the card or error message	40
power_off	$4D_H$	Acknowledge	32
power_up_iso	69_H	ATR from the card or error message	32
power_up_S9	$6B_H$	ATR from the card or error message	37
power_up_I2C	$6C_H$	Acknowledge or error message	40
power_up_1.8V	68_H	ATR from the card or error message	31
power_up_3V	$6D_H$	ATR from the card or error message	31
power_up_5V	$6E_H$	ATR from the card or error message	32

Command	Code	Answer from reader	page
idle_mode (clock stop low)	$A2_H$	Acknowledge	30
power_down_mode	$A3_H$	Acknowledge	30
idle_mode (clock stop high)	$A4_H$	Acknowledge	30
set_nad	$A5_H$	Acknowledge or error message	36
get_card_param	$A6_H$	Fi, Di, CLK, T of the card in use or error message	36
get_reader_status	AA_H	Information about the current state of the reader	28
power_up_S10	$C1_H$	ATR from the card or error message	39
process_S10	$C2_H$	Data read from the card or error message (read operation) Acknowledge or error message (write operation)	39
read_IO	CE_H	Value on the IO pins	42
set_IO	CF_H	Acknowledge	42

Table 6. Outgoing commands (only)

Command	Code	Parameter	Description
Card_take_off	$A0_H$	00_H	These commands are sent as soon as a card is inserted or extracted without any command coming from the system. These commands use the same operating code but the extra parameter gives the additional information. These outgoing commands are sent only when the host is waiting for a reply or is in stand by; when the card is extracted whereas the host is sending a frame to TDA8029, the card_take_off message will be sent from TDA8029 only when it has received the complete frame coming from the host controller. This system prevents any conflict on the serial line.
Card_insertion	$A0_H$	01_H	
Card deactivated	XX_H	$A1_H$	The card is deactivated due to a hardware problem (short on Vcc, overcurrent)
Time out	XX_H	FF_H	Time out problem on (TDA8029) Rx line This command is used in order to warn the host controller that the last communication has broken down (time out problem) so that the Rx line of TDA8029 does not remain blocked. The time out condition is a silence greater than 10 ms in the host command frame.
Frame lost	XX_H	$F1_H$	An unexpected host controller command frame has been received by the TDA8029 while it was busy to process a previous command frame.

In the last three commands, the code value is the previous code value used during a normal exchange.

8. Error list

The error list gives the status code identification and a brief signification of the status error code.

Table 7. List of error codes

Status code	Meaning
08_H	Length of the data buffer too short
20_H	Wrong APDU
21_H	Too short APDU
22_H	Card mute now (during T=1 exchange)
24_H	Bad NAD
26_H	Resynchronized
27_H	Chain aborted
29_H	Overflow from card
30_H	Non negotiable mode (TA2 present)
31_H	Protocol is neither T=0 nor T=1 (negotiate command)
32_H	T=1 is not accepted (negotiate command)
33_H	PPS answer is different from PPS request
34_H	Error on PCK (negotiate command)
35_H	Bad parameter in command
38_H	TB3 absent
39_H	PPS not accepted (no answer from card)
$3B_H$	Early answer of the card during the activation
40_H	Card Deactivated
55_H	Unknown command
80_H	Card mute (after power on)
81_H	Time out (waiting time exceeded)
83_H	Too much parity errors in reception
84_H	Too much parity errors in transmission
86_H	Bad FiDi
88_H	ATR duration greater than 19200 etus (E.M.V.)
89_H	CWI not supported (E.M.V.)
$8A_H$	BWI not supported (E.M.V.)

Status code	Meaning
$8B_H$	WI (Work waiting time) not supported (E.M.V.)
$8C_H$	TC3 not accepted (E.M.V.)
$8D_H$	Parity error during ATR
92_H	Specific mode byte TA2 with b5 byte=1
93_H	TB1 absent during a cold reset (E.M.V.)
94_H	TB1different from 00 during a cold reset (E.M.V.)
95_H	IFSC<10H or IFSC=FFH
96_H	Wrong TDi
97_H	TB2 is present in the ATR (E.M.V.)
98_H	TC1 is not compatible with CWT
99_H	IFSD not accepted
$9B_H$	Not T=1 card
$A0_H$	Procedure byte error
$B0_H$	Writing attempt in a protected byte (S9 cards)
$B1_H$	Pin Code error (S9 cards)
$B2_H$	Writing error (S9 cards)
$B3_H$	Too much data requested in a reading operation (S9 cards)
$B4_H$	Error counter protected (S9 cards)
$B5_H$	Writing attempt without Pin Code verification (S9 cards)
$B6_H$	Protected bit already set (S9 cards)
$B7_H$	Verify Pin Code error (S9 cards)
$C0_H$	Card absent
$C1_H$	I/O line locked while the TDA8029 attempts to access to an I2C or S10 card
$C3_H$	Checksum error
$C6_H$	ATR not supported
CC_H	No acknowledge from the I2C synchronous card
CD_H	Generic error during an exchange with an I2C synchronous card
$E1_H$	Card clock frequency not accepted (after a set_clock_card command)
$E2_H$	UART overflow
$E3_H$	Supply voltage drop-off

Status code	Meaning
$E4_H$	Temperature alarm
$E5_H$	Card deactivated
$E9_H$	Framing error
$F0_H$	Serial LRC error
$F1_H$	At least one command frame has been lost
FF_H	Serial time out

Table 8. Error codes for each command

Command		Possible returned error code
Power UP 1.8V, 3V, 5V		31h, 35h, 38h, 3Bh, 80h, 85h, 86h, 88h, 89h, 8Ah, 8Bh, 8Ch, 8Dh, 92h, 93h, 94h, 95h, 96h, 97h, 98h, C0h, C3h, C4h, C6h, C7h, E2h, E3h, E4h, E9h, F0h, F1h, FFh
Power up in ISO mode		31h, 35h, 3Bh, 80h, 96h, C0h, C3h, C4h, C6h, C7h, E2h, E3h, E4h, E9h, F0h, F1h, FFh
Card Command	T=0	08h, 20h, 21h, 0x40, A1h, 81h 83h, 84h, 91h, A0h, C0h, E2h, E3h, E4h, E9h, F0h, F1h, FFh
	T=1	08h, 22h, 24h, 25h, 26h, 27h, 28h, 29h, 0x40, A1h, 83h, 90h, C0h, E2h, E3h, E4h, E9h, F0h, F1h, FFh
Negotiate		30h, 31h, 33h, 34h, 35h, 39h, 0x40, A1h, C0h, E2h, E3h, E4h, E9h, F0h, F1h, FFh
Set Clock Card		C0h, E1h, F0h, F1h, FFh
Set card baud rate		86h, C0h, F0h, F1h, FFh
Set Nad		24h, F0h, F1h, FFh
Get card parameters		0x40, A1h, C0h, F0h, F1h, FFzh
Ifsd request		0Ah, A1h, 9Bh, C0h, E2h, E3h, E4h, E9h, F0h, F1h, FFh
Send mask number		F0h, F1h, FFh
Check presence card		F0h, F1h, FFh
Set serial baud rate		55h, F0h, F1h, FFh
Power off		F0h, F1h, FFh
Idle mode clock stop low and high		55h, F0h, F1h, FFh
Get reader status		F0h, F1h, FFh
Power down mode		55h, F0h, F1h, FFh
Read IO and Set IO		F0h, F1h, FFh

Command	Possible returned error code
Power up I2C	C0h, F0h, F1h, FFh
Read I2C	C0h, C1h, CCh, CDh, F0h, F1h, FFh
Read I2C extended	C0h, C1h, CCh, CDh, F0h, F1h, FFh
Read current I2C	C0h, C1h, CCh, CDh, F0h, F1h, FFh
Write I2C	55h, C0h, C1h, CCh, CDh, F0h, F1h, FFh
Power up S9	C0h, F0h, F1h, FFh
Read S9	B3h, C0h, F0h, F1h, FFh
Read S9 protection	B3h, C0h, F0h, F1h, FFh
Write S9 unprotected	55h, B0h, B2h, B5h, C0h, F0h, F1h, FFh
Write S9 protected	55h, B0h, B2h, B5h, C0h, F0h, F1h, FFh
Verify PIN S9	55h, B1h, B4h, B7h, C0h, F0h, F1h, FFh
Compare S9	55h, B6h, B7h, C0h, F0h, F1h, FFh
Power up S10	C0h, F0h, F1h, FFh
Process command S10	C0h, C1h, F0h, F1h, FFh

9. Commands description

9.1 General commands

9.1.1 send_num_mask

This command is used to identify the software version which is masked in TDA8029 ROM.

For example the current software will be coded as: “06 Release 1.1” (14 ASCII characters)

System to TDA8029: 60 00 00 0A 6A

TDA8029 to System: 60 00 0E 0A 30 36 20 52 65 6C 65 61 73 65 20 31 2E 31 0E

9.1.2 check_card_presence

This command is used to check the presence of a card.

System to TDA8029: 60 00 00 09 69

TDA8029 to System: 60 00 01 09 PRES LRC

Where PRES indicates the presence of a card (00 if there is no card, 01 if a card is present).

9.1.3 get_reader_status

This command is used to check the status of the reader.

System to TDA8029: 60 00 00 AA CA

TDA8029 to System: 60 00 01 AA STATUS LRC

Where the latched state of the TDA8029 is given in STATUS byte.

nu	nu	nu	nu	SUPL	PROTL	PTL	PRES

PRES card presence (0: card absent, 1: card present)

PTL overheating detection

PROTL default detected on card reader (protection on VCC or RST)

SUPL supervisor activation

The byte STATUS is cleared (except PRES bit) after having launch this command.

9.1.4 set_serial_baud_rate

This command is used for changing the baud rate onto the serial link between the host and the interface card. The default value is set to 38400 baud.

A parameter has to be transmitted in order to choose the baud rate:

System to TDA8029: 60 00 01 0D PAR LRC
TDA8029 to System: 60 00 00 0D 6D

Table 9. Baud rate parameter

Baud rate (Baud)	Parameter
4800	00
9600	01
19200	02
38400	03
57600	04
76800	05
115200	06

After a baud rate change, the new value takes place for the next command sent by the host.

9.1.5 set_esm_properties

This command is used to fix the behavior of the TDA8029 if the Energy Saving Mode is activated.

By default, the clock stop information contained in the ATR of the activated card is used during the session with this card to set the clock when the TDA8029 enters in power down mode. Thus, if the card does not explicitly indicate that it supports clock stop mode (High or Low), the card clock will be set to Fint/2.

To save even more energy during these periods, the clock mode can be forced using this command.

System to TDA8029: 60 00 02 BC STOP STATE LRC

TDA8029 to System: 60 00 00 BC DC

Where **STOP** indicates the clock stop request:

00 to set the clock according to card indications *(default behavior)*

01 to force the clock stopping

STATE is the clock stop level if requested (when STOP = 01):

00 to stop clock LOW

01 to stop clock HIGH

Note 1: This command can be used only when the Energy Saving Mode is activated. If launched when the Energy Saving Mode is not activated, an *UNKNOWN_COMMAND* error will be returned by the TDA8029.

Note 2: Once this command has been launched to the TDA8029, all the further activations of cards will follow the behavior defined within this command. One has to use this command again to change the behavior, e.g. to come back to a clock at Fint/2. Furthermore, even if this command is used to force a clock mode, when a card with defined clock stop conditions is encountered, the clock stop mode indicated in the card's ATR will be used.

9.1.6 time_out

This command is sent from TDA8029 to the host controller if, during a transmission from the host controller to TDA8029, the time interval between 2 characters exceeds 10ms. This timing is calculated between each character of a frame, starts after the first character, and is disabled after the last character of the frame. This feature has been implemented in order to avoid any blocking of the transmission line between the host controller and TDA8029.

TDA8029 to System: E0 00 01 6F FF 71

9.1.7 idle_mode (clock stop low)

This command is used to set the controller in idle mode. The card, if activated, has its clock (CLK) set to low level but is still active.

Any command from the host on the serial line will wake up the device.

System to TDA8029: 60 00 00 A2 C2

TDA8029 to System: 60 00 00 A2 C2

idle_mode (clock stop high)

This command is used to set the controller in idle mode. The card, if activated, has its clock (CLK) set to high level but is still active.

Any command from the host on the serial line will wake up the device.

System to TDA8029: 60 00 00 A4 C4

TDA8029 to System: 60 00 00 A4 C4

9.1.8 power_down_mode

This command is used to set the controller in power down mode; if the card is active, it is then deactivated. Exiting this mode is possible with a hardware reset of TDA8029 or an external interruption (INT0, INT1 or Rx).

System to TDA8029: 60 00 00 A3 C3

TDA8029 to System: 60 00 00 A3 C3

9.2 Asynchronous card related commands

9.2.1 power_up commands

There are four different power up commands (5V, 3V, 1.8V or ISO). Two of them (power_up_3V and power_up_5V) have to be followed by a parameter:

- 00_H indicates that all the parameters of the ATR of the card compliant with ISO7816-3 will be taken into account.
- 01_H indicates that only the ATR of cards whose parameters are inside the E.M.V. 4.0 specification scope will be taken into account; cards having an ATR which does not comply with E.M.V. 4.0 requirements will be rejected.

9.2.1.1 power_up_5V

This command allows to activate the card at a VCC of 5V. All the signals going to the card will be referenced to this VCC.

An activation sequence is processed following the ISO7816-3 normalization (VCC is rising, I/O is enabled, CLK is started, and RST is processed). If the card answers to this command, the answer will content all the ATR parameters; these parameters are memorized in TDA8029 and will be taken into account during the whole card session (till the card is deactivated or till a warm reset is processed). The structure of the answer is the following:

System to TDA8029:

60	00 01	6E	01	0F
ACK	length	code	E.M.V.	LRC

TDA8029 to System:

60	XX XX	6E	nnnnnnnnnnnnnnnnnnn	ZZ
ACK	length	code	ATR parameters	LRC

019aab415

Fig 25. power_up_5V frame exchange

If the card is in specific mode, TDA8029 will process the next command directly using the new interface parameters of this specific mode. If the card proposes a different Fi/Di in the ATR than the default value (Fi/Di=372), it is up to the application to make a PPS command by using the negotiate command. If the card proposes 2 different protocols in its ATR, it is up to the application to make a PPS command by using the negotiate command.

If the card does not answer to the reset, a status giving an error code is returned to the application.

In the case of E.M.V. compliant power up, if the card is using T=1 protocol, just after having received the ATR, TDA8029 sends an IFSD request to the card indicating that the reader can manage a data buffer of 254 bytes (FEH).

The power_up_5V command can be used to generate a warm reset if the card is already activated.

9.2.1.2 power_up_3V

This command allows to activate the card at a VCC of 3V. Every signal going to the card will be referenced to this VCC.

See power_up_5V for the other characteristics.

9.2.1.3 power_up_1.8V

This command allows to activate the card at a VCC of 1.8V. Every signal going to the card will be referenced to this VCC.

See power_up_5V for the other characteristics.

9.2.1.4 power_up_iso

This command does not need any argument. The principle consists to activate the card as described in ISO 7816-3:

- attempt to activate the card at a VCC of 3V, if the cards answers correctly and if it indicates in its ATR that it is a class A or a class AB card (TAi with T=15), then the command is finished and the ATR is returned to the host,
- if in the previous stage, the card did not answer correctly or did not specify in its ATR that it was a class A or a class AB card, a new activation of the card is launched at 5V. If the card does not answer to the reset, a status giving an error code is returned to the application, otherwise the answer contains all the parameters of the card.

See power_up_5V for the other characteristics (when parameter of the command is ISO, not E.M.V.).

9.2.2 power_off

This command is used to deactivate the card whatever it has been activated for 3V or 5V operation. A deactivation sequence is processed following the ISO 7816-3 normalization in about 100µs.

System to TDA8029: 60 00 00 4D 2D

TDA8029 to System: 60 00 00 4D 2D

9.2.3 card_command (APDU)

This command is used to transmit card commands under APDU format from system to TDA8029 whatever T=0 or T=1 protocol are used. Short or extended commands (see limitations in chapter 10.1) can be used.

An answer to such a command is also made in APDU format from TDA8029 to the system.

Example:

System to TDA8029: 60 00 07 00 00 A4 00 00 02 4F 00 8E

TDA8029 to System: 60 00 02 00 90 00 F2

9.2.4 process_T=1_command

This command may be used if the application layer provides the complete T=1 frame including prologue, information and epilogue fields. If it is not the case, the above `card_command` opcode shall be used.

This command is used from the application layer in order to send a complete T=1 frame to the card. This command includes the specific framing used in T=1 protocol (Prologue Field, Information Field, Epilogue Field) and will be sent transparently to the card. The answer from the card will be sent as a complete T=1 frame to the application layer. The internal timing of a block (Character Waiting Time) will be handled by TDA8029. The block Waiting Time will also be controlled by TDA8029. In case of Waiting Time Extension request (WTX) from the card, it will be taken into account by the TDA8029.

System to TDA8029	60	XX XX	01	NAD PCB LEN A_1 A_2 …… A_N EDC	LRC
TDA8029 to System	60	00 06	01	NAD PCB LEN SW1 SW2 EDC	LRC

Where A1 A2…..An is information field sent to the card

XX XX is the length of the frame from NAD to EDC

In case of chaining:

System to TDA8029	60	XX XX	01	NAD 20 LEN A_1 A_2 …… A_N EDC	LRC
TDA8029 to System	60	00 04	01	NAD 90 00 EDC	LRC
System to TDA8029	60	YY YY	01	NAD 40 LEN A_{N+1} A_{N+2} … … A_Z EDC	LRC
TDA8029 to System	60	ZZ ZZ	01	NAD PCB LEN D_1 D_2 … … D_N EDC	LRC

9.2.5 negotiate

This command is used to make a PPS (Protocol and Parameter Selection) to the card, if in its ATR the card proposes a different Fi/Di or 2 different protocols. By using this command a PPS will be made to the card with the Fi or Di and protocol type entered as a parameter (PP). It is up to the host to make the correct Fi/Di submission to the card.

Example:

System to TDA8029: 60 00 02 10 PP FD LRC

TDA8029 to System: 60 00 00 10 70

Where FD is the ratio Fi/Di given by TA1 parameter of the ATR and PP is the protocol to be used.

If the command is acknowledged, any subsequent exchanges between the card and TDA8029 will be made by using the new parameters.

9.2.6 Ifsd_request

This command is used to send a S(IFS request) block to the card indicating the maximum length of information field of blocks which can be received by the interface device in T=1 protocol. The initial size following the answer to reset is 32 bytes and this size shall be used throughout the rest of the card session or until a new value is negotiated by the terminal by sending a S(IFS request) block to the card.

In E.M.V. mode, the IFSD size is automatically negotiated to 254 just after the ATR has been received.

System to TDA8029: 60 00 01 0C PAR LRC

TDA8029 to System: 60 00 00 0C 6C

Where PAR is the IFSD size.

9.2.7 set_clock_card

This command is used for changing the card clock frequency. The default value is set to FXTAL/4 which is 3.68625 MHz.

A parameter has to be transmitted in order to choose the card clock frequency:

System to TDA8029: 60 00 01 11 PAR LRC

Table 10. set_clock_card parameter
Based on a crystal with a frequency equal to 14.745MHz

Frequency	Parameter
Fxtal =14.745MHz	00
Fxtal/2=7.37MHz	02
Fxtal/4=3.68MHz	04
Fxtal/8=1.84MHz	06

After a card clock frequency change, all the waiting times are internally set to the new value.

Before applying the requested clock, the compatibility of the frequency with the current Fi used by the card is checked as described in ISO7816-3. For example, if the card has answered in its ATR a Fi parameter of 372 or 558 (fmax ≤ 6MHz), a change of the card clock frequency to Fxtal (14.745MHz) or Fxtal/2 (7.37MHz) will not be processed and an error status will be sent to the application.

9.2.8 card_take_off and card_insertion

These two commands are sent directly to the system processor as soon as a card extraction or insertion has occurred.

TDA8029 to System: 60 00 01 A0 00 C1 for a card extraction

60 00 01 A0 01 C0 for a card insertion.

9.2.9 set_card_baud_rate

This command is used mainly for cards which are not fully ISO 7816-3 compliant with specific and negotiable modes. As a matter of fact some cards are in specific mode but they do not give TA2 parameter in their answer to reset. So the UART has to be set to the right baud rate by means of this specific command which programs the baud rate. For non ISO baud rates there is a possibility to increase the capability of the reader by setting the bit CKU which divides by 2 the number of clock cycles of the etu and thus doubles the baud rate of the ISO UART.

Example:

System to TDA8029: 60 00 02 0B XX CKU LRC

TDA8029 to System: 60 00 00 0B LRC

Where XX is the value of FiDi

if CKU=0, the baud rate is defined by FiDi

if CKU=1, the baud rate is 2 * the baud rate is defined by FiDi

For an etu of 372 clock cycles: XX=FiDi=0x11

prescaler = 31, divider = 12; 31 * 12 = 372, CKU=0.

Table 11. Mask06 and Mask07 supported baudrates

As the baud rates in dark boxes are using CKU bit, they are not reachable when CLK = Xtal

TA1	CLK/ETU	TA1	CLK/ETU	TA1	CLK/ETU	TA1	CLK/ETU	TA1	CLK/ETU
0x01	372	0x31	744	0x54	186	0x95	32	0xC1	1536
0x02	186	0x32	372	0x55	93	0x96	16	0xC2	768
0x03	93	0x33	186	0x56	46.5	0xA1	768	0xC3	384
0x04	46.5	0x34	93	0x58	124	0xA2	384	0xC4	192
0x08	31	0x35	46.5	0x61	1860	0xA3	192	0xC5	96
0x11	372	0x38	62	0x62	930	0xA4	96	0xC6	48
0x12	186	0x41	1116	0x63	465	0xA5	48	0xC8	128
0x13	93	0x42	558	0x64	232.5	0xA8	64	0xD1	2948
0x14	46.5	0x43	279	0x68	155	0xB1	1024	0xD2	1024
0x18	31	0x44	139.5	0x69	93	0xB2	512	0xD3	512
0x21	558	0x48	93	0x91	512	0xB3	256	0xD4	256
0x22	279	0x51	1488	0x92	256	0xB4	128	0xD5	128
0x23	139.5	0x52	744	0x93	128	0xB5	64	0xD6	64
0x28	46.5	0x53	372	0x94	64	0xB6	32		

9.2.10 set_nad

This command is used from the application layer in order to specify a SAD (source address) and a DAD (destination address) for a logical connection using T=1 protocol as defined in ISO7816-3. The default value is 00 and will be kept until the send NAD command has been notified to the TDA8029. Any NAD submission where SAD and DAD are identical (except 00) will be rejected. If bits b4 or b8 of the NAD required are set to 1 (VPP programming) the NAD will be rejected.

The NAD shall be initialized before any information exchange with the card using T=1 protocol, otherwise and error message will be generated.

System to TDA8029: 60 00 01 A5 NAD LRC

TDA8029 to System: 60 00 00 A5 LRC

Where NAD is the new value of NAD immediately taken into account.

9.2.11 get_card_param

This command is used from the application level in order to get the Fi and Di parameters of the card in use, the current card clock frequency, and the protocol in use.

FiDi parameter will be given on one byte (FiDi), the card clock frequency on one byte (CC), and the protocol on one byte (TT).

FiDi will give the value of the current Fi Di (Example 11H for Fi=372 and Di=1)

CC will take the value of the 4 lowest bits of CCR register.

TT will take value 00H for protocol T=0 and value 01H for protocol T=1.

If there is no card in use, an error message will be generated.

System to TDA8029: 60 00 00 A6 C6

TDA8029 to System: 60 00 03 A6 FiDi CC TT LRC

Where:

FIDI gives the current FIDI coded as in TA1 parameter,

CC gives the value of the card clock frequency as coded in CCR register of TDA8029,

TT gives the protocol used by the card (00 for protocol T=0, 01 for protocol T=1).

9.3 Synchronous card related commands

9.3.1 Synchronous card S=9

9.3.1.1 power_up_S9

The card is powered under 5V and answers 4 bytes as Answer To Reset.

System to TDA8029: 60 00 00 6B 0B

TDA8029 to System: 60 00 04 6B XX_1 XX_2 XX_3 XX_4 LRC

Where XX_1 XX_2 XX_3 XX_4 are the data sent by the card in its ATR.

The card is then ready to operate.

9.3.1.2 read_S9

This command allows to read bytes of 8 bits in the card from the specified address.

System to TDA8029: 60 00 04 03 AD_H AD_L NB_H NB_L LRC

TDA8029 to System: 60 NB_H NB_L 03 D_1 D_2 D_3…D_n LRC

Where AD_H AD_L indicates the address where to read (coded on 2 bytes)

NB_H NB_L is the number of bytes to read (coded on 2 bytes)

D_1 D_2 $D_{3...}D_n$ are the NB_H NB_L data read

9.3.1.3 read_S9_protection

This command allows to read bytes of 8 bits + the protect bit as the 9th bit in the card from the specified address.

System to TDA8029: 60 00 04 04 AD_H AD_L NB_H NB_L LRC

TDA8029 to System: 60 (NB_H NB_L)*2 04 D_1 0/1 D_2 0/1 D_3 0/1…D_n 0/1 LRC

Where AD_H AD_L indicates the address where to read (coded on 2 bytes)

NB_H NB_L is the number of bytes to read (coded on 2 bytes)

D_1 0/1 D_2 0/1 D_3 0/1 ...D_n 0/1 are the NB_H NB_L data read

The process is the same as for the command read_8bit_S9 except that the value of the protect bit is added in the answer.

Each byte read is followed by one byte that informs if the byte is protected or not (0x00: protected, 0x01 not protected).

9.3.1.4 write_S9_protected

This command allows to write bytes with protected bit as 9th bit from the specified address.

System to TDA8029: 60 NB_H NB_L 05 AD_H AD_L D1 D2 D3 Bn LRC

TDA8029 to System: 60 00 00 05 LRC

Where	AD_H AD_L	indicates the address where to write (coded on 2 bytes)
	(NB_H NB_L)-2	is the number of bytes to write (coded on 2 bytes)
	D1 D2 D3…Dn	are the data to write in the card

9.3.1.5 write_S9_unprotected

This command allows to write bytes without protection from the specified address.

System to TDA8029: 60 NB_H NB_L 06 AD_H AD_L D1 D2 D3 Dn LRC

TDA8029 to System: 60 00 00 06 LRC

Where	AD_H AD_L	indicates the address where to write (coded on 2 bytes)
	(NB_H NB_L)-2	is the number of bytes to write (coded on 2 bytes)
	D1 D2 D3…Dn	are the data to write in the card

9.3.1.6 verify_pin_code

System to TDA8029: 60 00 03 07 XX PIN1 PIN2 LRC

TDA8029 to System: 60 00 00 07 LRC

Where	XX	in the bit mask for error counter
	PIN1	is the first PIN CODE
	PIN2	is the second PIN CODE

9.3.1.7 compare

System to TDA8029: 60 00 03 08 AD_H AD_L XX LRC

TDA8029 to System: 60 00 00 08 LRC

Where	AD_H AD_L	indicates the address of byte to compare
	XX	is the byte to compare

9.3.2 Card S=10

9.3.2.1 power_up_S10

This command powers up the S10 card; 4 bytes of Answer To Reset from the card are expected.

System to TDA8029: 60 00 00 C1 A1

TDA8029 to System: 60 00 04 C1 xx_1 xx_2 xx_3 xx_4 LRC

Where xx_1 xx_2 xx_3 xx_4 are the data sent by the card in its ATR.

The card is then ready to operate.

9.3.2.2 process_S10

This command allows either to read or to write bytes from or into an S10 card from the specified address.

<u>In case of a read command</u>:

System to TDA8029: 60 00 03 C2 CB AD NB LRC

TDA8029 to System: 60 NB_H NB_L 12 D1 D2 D3…Dn LRC

Where	CB	is the control byte
	AD	is the address byte
	NB	is the number of bytes to read
	D1 D2 D3…Dn	are the NB data read

<u>In case of a write command</u>:

System to TDA8029: 60 ML_H ML_L C2 CB AD D1 D2 D3…Dn LRC

TDA8029 to System: 60 00 00 C2 A2

Where	ML_H ML_L	is the total message length
	CB	is the control byte
	AD	is the address byte
	D1 D2 D3…Dn	are the data to write in the card

9.3.3 I2C cards

9.3.3.1 power_up_I2C

This command powers up the I2C card; no data are expected from the card.

System to TDA8029: 60 00 00 6C 0C

TDA8029 to System: 60 00 00 6C 0C

The card is then ready to operate.

9.3.3.2 read_I2C

This command allows to read bytes from the specified address in a standard I2C card.

System to TDA8029: 60 00 05 12 I2CAd AD_H AD_L 00 NB LRC

TDA8029 to System: 60 NB_H NB_L 12 xx xx xx xx xx xx xx xx xx LRC

Where	I2CAd	is the physical I2C address of the embedded component
	AD_H AD_L	indicates the address where to read (coded on 2 bytes)
	NB	is the number of bytes to read (coded on 1 byte)
	xx xx xx	are the NB_H NB_L data read

Remark: Normally ADH should be fixed to 00h since this command is used with standard I2C cards.

The field ADH has been added to manage multiple blocks memory found in some I2C cards (Gemplus GFM4K for example). In that case, ADH contains the number of the page to read (00 H or 01H).

9.3.3.3 read_I2C_extended

This command allows to read bytes from the specified address in an extended I2C card.

System to TDA8029: 60 00 05 13 I2CAd AD_H AD_L NB_H NB_L LRC

TDA8029 to System: 60 NB_H NB_L 13 xx xx xx xx xx xx xx xx xx LRC

Where	I2CAd	is the physical I2C address of the embedded component
	AD_H AD_L	indicates the address where to read (coded on 2 bytes)
	NB_H NB_L	is the number of bytes to read (coded on 2 bytes)
	xx xx xx	are the NB_H NB_L data read

9.3.3.4 read_current_I2C

This command allows to read bytes from the current address in a standard I2C card.

System to TDA8029: 60 00 03 23 I2CAd NB_H NB_L LRC

TDA8029 to System: 60 NB_H NB_L 23 D1 D2 D3…Dn LRC

Where	I2CAd	is the physical I2C address of the embedded component
	NB_H NB_L	is the number of bytes to read (coded on 2 bytes)
	D1 D2 D3…Dn	are the NB_H NB_L data read

9.3.3.5 write_I2C

This command allows to write bytes in an I2C card from a specified address.

Two different cases can be met depending on the type of the I2C card: using the extended mode or not.

If the card is using extended mode, then the address is coded on 2 bytes, as if the card is not using the extended mode, the address is coded only on one byte. This is the responsibility of the application layer to know if the current card is using the extended mode or not.

<u>Card using extended mode</u>:

System to TDA8029: 60 NB_H NB_L 02 I2CAd AD_H AD_L D1 D2 D3 … Dn LRC

TDA8029 to System: 60 00 00 02 LRC

Where	I2CAd	is the physical I2C address of the embedded component
	AD_H AD_L	indicates the address where to write (coded on 2 bytes)
	(NB_H NB_L)-3	is the number of bytes to write (coded on 2 bytes)
	D1 D2 D3…Dn	are the data to write in the card

<u>Card not using extended mode</u>:

System to TDA8029: 60 NB_H NB_L 02 I2CAd AD D1 D2 D3 … Dn LRC

TDA8029 to System: 60 00 00 02 LRC

Where	I2CAd	is the physical I2C address of the embedded component
	AD	indicates the address where to write (coded on 1 byte)
	(NB_H NB_L)-2	is the number of bytes to write (coded on 2 bytes)
	D1 D2 D3…Dn	are the data to write in the card

Remark: This function does not manage the change of segment in the EEPROM. The maximum length of the data stream that can be programmed in one step depends of the embedded component.

9.4 General purpose IO commands

9.4.1 read_IO

This command is used to read the current state of the four general purpose IO of the TDA8029.

System to TDA8029: 60 00 00 CE AE

TDA8029 to System: 60 00 01 CE VAL LRC

VAL is coded as follows:

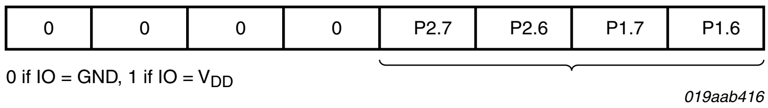


Fig 26. Read IO command – VAL byte description

9.4.2 set_IO

This command is used to set one of the general purpose IO of the TDA8029 to a specified logic level.

System to TDA8029: 60 00 02 CF IO VAL LRC

TDA8029 to System: 60 00 00 CF AF

Where IO and VAL are coded as follows:

Table 12. IO coding in set_IO command

IO	Port
0x01	P1.6
0x02	P1.7
0x03	P2.6
0x04	P2.7

Table 13. VAL coding in set_IO command

VAL	Level to apply
0x00	GND
0X01	VDD

10. Information field for asynchronous cards

The data buffer has a size of 512 bytes whose 6 bytes located at the end of the buffer are used by the internal library; so the data buffer has a real size of 506 bytes.

The information field that can include up to 506 bytes is composed of APDUs (Application Protocol Data Unit) according to the ISO7816-4 normalization definition.

Different examples are given according to Annex A of the E.M.V.'96 in T = 0.

TAL (System)		TTL (TDA8029)
Case 1 command		
{60, 00, 04, 00, CLA, INS, P1, P2, LRC} (CLA, INS, P1, P2: **4 header bytes**)	⇒	
	⇐	**{60, 00, 02, 00, 90, 00, LRC}**
Case 2 command		
{60, 00, 05, 00, CLA, INS, P1, P2, 00, LRC}	⇒	
	⇐	**{60, Licc+2, 00, [Data (Licc)], 90, 00, LRC}**
Case 3 command		
{60, Lc+5, 00, CLA, INS, P1, P2, Lc, [data Lc], LRC}	⇒	
	⇐	**{60, 00, 02, 00, 90, 00, LRC}**
Case 4 command		
{60, Lc+5+1, 00, CLA, INS, P1, P2, Lc, [data Lc], 00, LRC}	⇒	
	⇐	**{60, Licc+2, 00, [data Licc], 90, 00, LRC}**
Case 2 command **using the 61 and 6C procedure byte**		
Le = Licc or Le ≥ Licc		
{60, 00, 05, 00, CLA, INS,P1, P2, 00, LRC}	⇒	
	⇐	**60, D1+D2+Dn+2, 00, [data D1+D2+Dn], 90, 00, LRC}**

10.1 Extended cases

In T=0 protocol, the extended cases for APDUs are not supported on this mask.

In T=1 protocol, the use of the extended cases for APDUs is transparent from the host point of view as explained below as the TPDUs are identical to the APDUs.

Case 2 extended example:

APDU: CLA INS P1 P2 00 B2 B3 where B2 B3 is the length coded on 2 bytes (from 1 to 65535). With both mask 06 and mask 07 release, B2 B3 shall never exceed 498 bytes.

System to TDA8029: 60 00 07 00 CLA INS P1 P2 00 B2 B3 LRC

TDA8029 to card: NAD PCB 07 CLA INS P1 P2 00 B2 B3 EDC

Card to TDA8029: NAD PCB LEN1 D1 D2 …. Di EDC

Where LEN1 is related to the negotiated data buffer size.

TDA8029 to card: Rblock for acknowledge.

Card to TDA8029: NAD PCB LEN2 Di+1 D1+2 …. Dn SW1 SW2 EDC

Where n = B2 B3

For this example it is supposed that only one chaining step is necessary.

TDA8029 to System: 60 B4 B5 00 D1 D2 …….. Dn SW1 SW2 LRC

(B4 B5 =n+2)

References: ISO 7816-4 §5.3 and Annex B.

11. Conclusion

The following features give the general characteristics of both mask 06 and mask 07:

- 1.8V, 3V and 5V cards supported
- E.M.V. 4.0 validated but possibility to switch to full ISO 7816-3
- Data buffer up to 506 bytes
- Asynchronous protocols (T=0 and T=1) supported
- A I2C, S9, S10 synchronous cards supported
- Serial link for control and communication with variable baud rates from 4800 to 115200 baud
- Automatic hardware protections in the event of card take off, supply voltage drop short circuit or overheating
- All ISO7816-3 baud rates supported on the I/O line
- Possible selection of card clock frequencies
- Communication with the host made at the APDU level (asynchronous cards) or also possible at TPDU level for protocol T=1
- Single +2.7V to +6.0V supply voltage
- Settings of switches for the configuration:

Table 14. Host interface hardware configuration

Interface	P17 (#1)
RS232 (see section 5)	VDD
I2C (see section 6)	GND

Table 15. ESM hardware configuration

ESM	P26 (#25)
ON	VDD or not connected
OFF	GND

12. ANNEX I: SCRTESTER

SCRTester is a PC software allowing to communicate with a Philips smart card reader (CAKE8029_11D for instance) through an RS232 serial link.

SCRTester can be used when the TDA8029 mask 06 or mask 07 is configured to be interfaced with a host controller by using a RS232 serial link

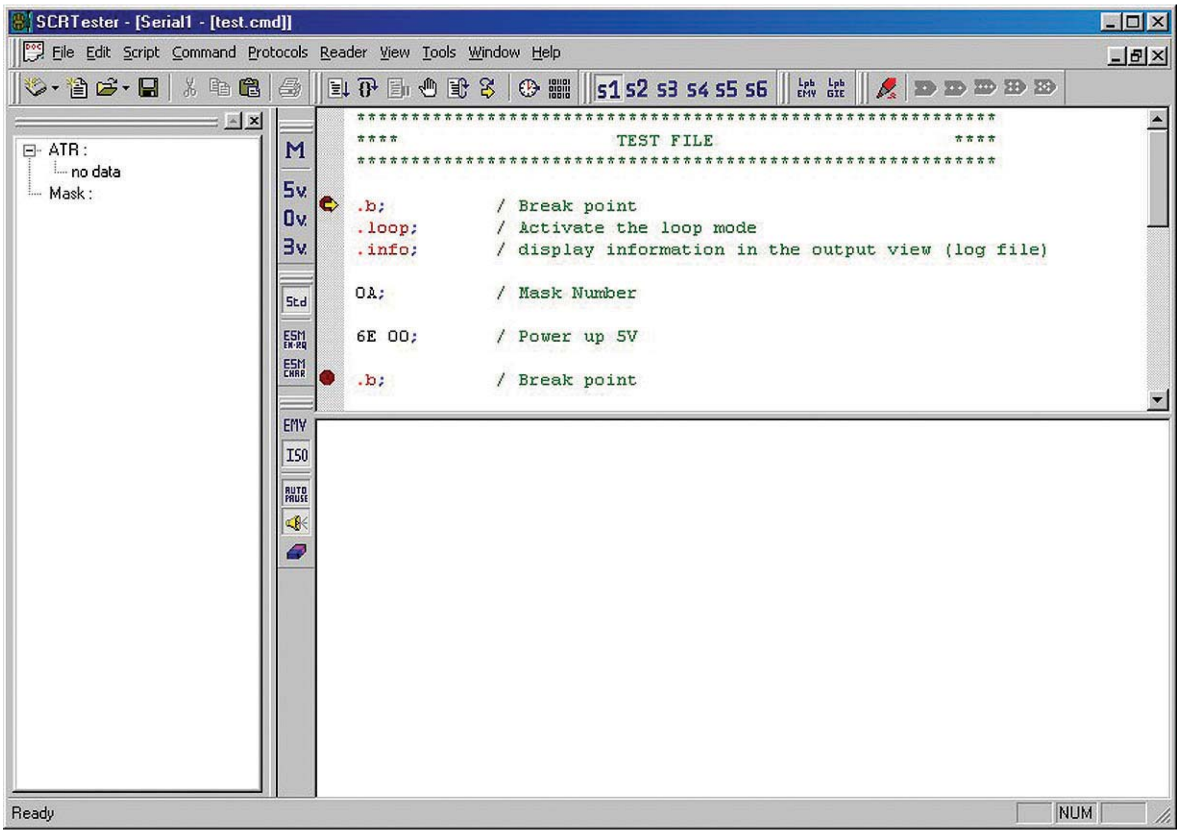
12.1 Installation

SCRTester is supplied in two floppy disks. Run the *setup.exe* file located on the floppy #1 to install SCRTester on your computer, and then follow the given indications.

Once installed, SCRTester is available in C:\Program Files\Philips Semiconductors directory (*SCRTester.exe*).

12.2 Run SCRTester

When SCRTester is launched for the first time, the following screen appears:



019aab417

Fig 27. SCRTester main window

SCRTester includes a complete help file that can be launched by pressing F1 function key or by selecting the *Help→Contents* menu item.

SCRTester tries to establish a serial connection with the reader on COM1 port. If this port is not available, the following warning message appears and then the user has to manually configure the port used by the reader by using the Reader menu item (select the correct port COMx and after that use the Connect command).

Fig 28. SCRTester – Serial port connection error window

12.2.1 The top right window

The top right window contains the command script file, which can be directly modified by the user.

The commands have to be written following the correct format defined in this application note. By default, SCRTester is configured in Command mode (Script menu item), i.-e. ALPAR header frame (except Command byte) and LRC character are not needed.

For example, to send a *send_num_mask* command to the reader (p. 28), one has to write

0A;

In the script window, SCRTester will automatically send the complete frame to the reader:

60 00 00 0A 6A

12.2.2 The bottom right window

The bottom right window contains all the commands sent to the reader (in red color) and the received answers (in blue color).

12.2.3 The left window: card parameters

In case of *send_num_mask* or *power_up* commands, the left window is refreshed with received information from the reader:

- The current mask number string is displayed,
- The complete ATR is decomposed into individual fields.

13. ANNEX II: Hardware information

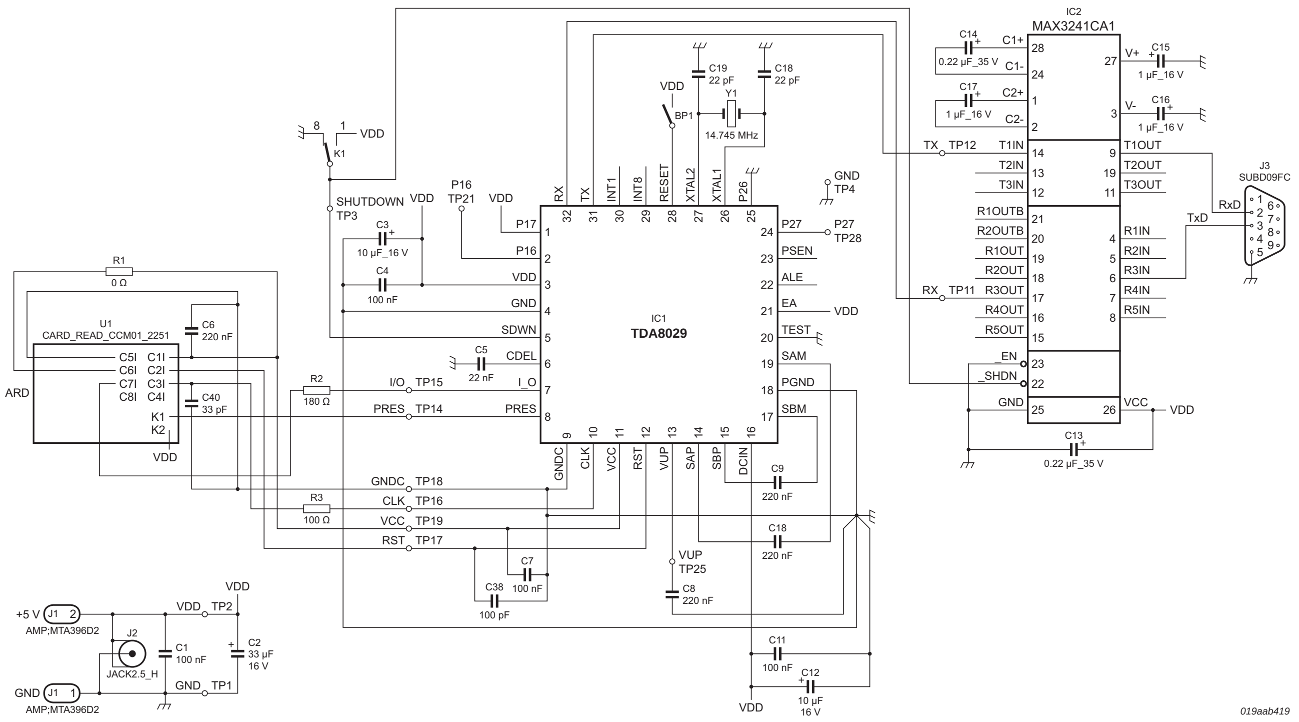


Fig 29. Design example - Electrical schematic

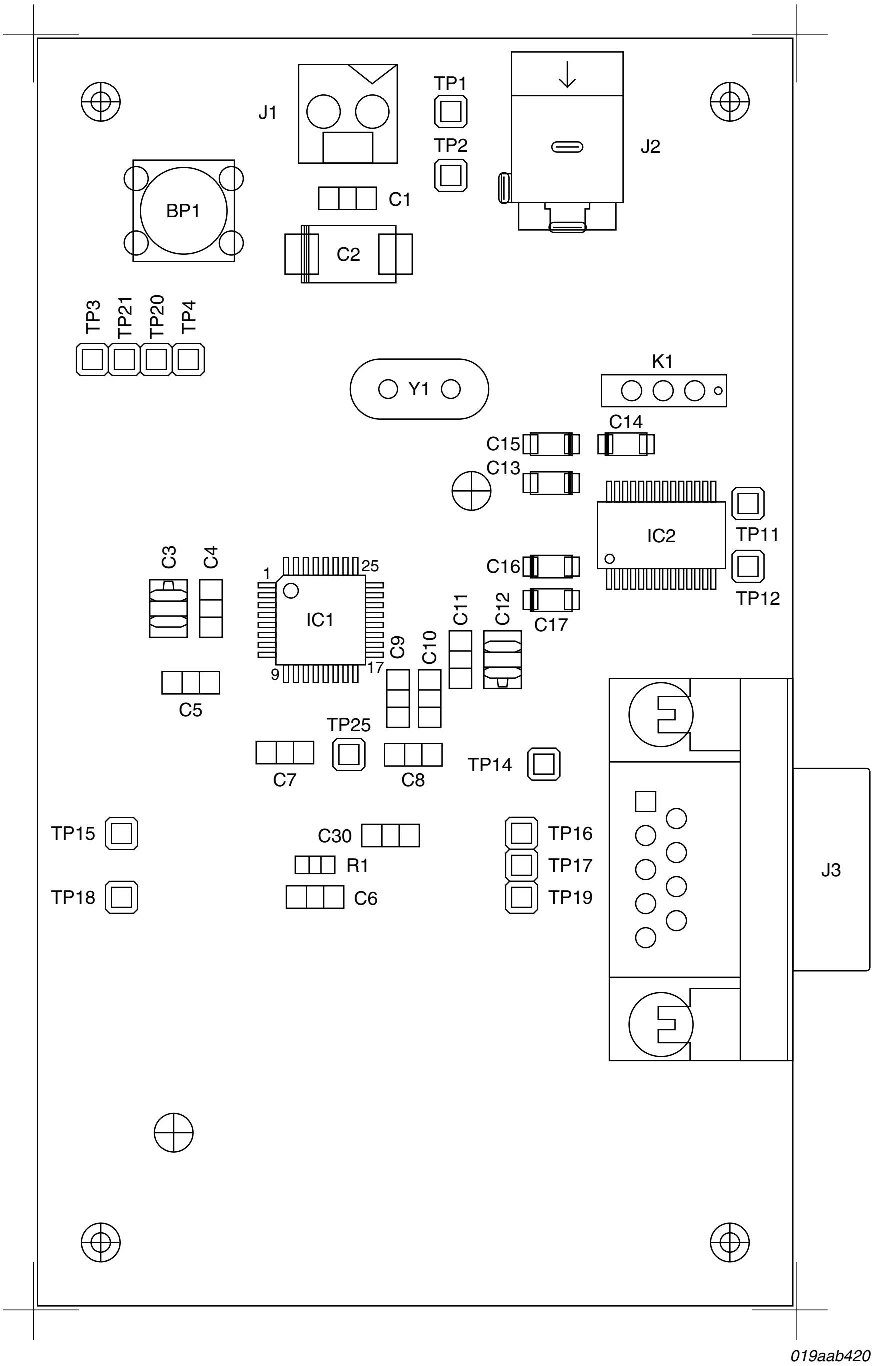


Fig 30. Design example - Components top view

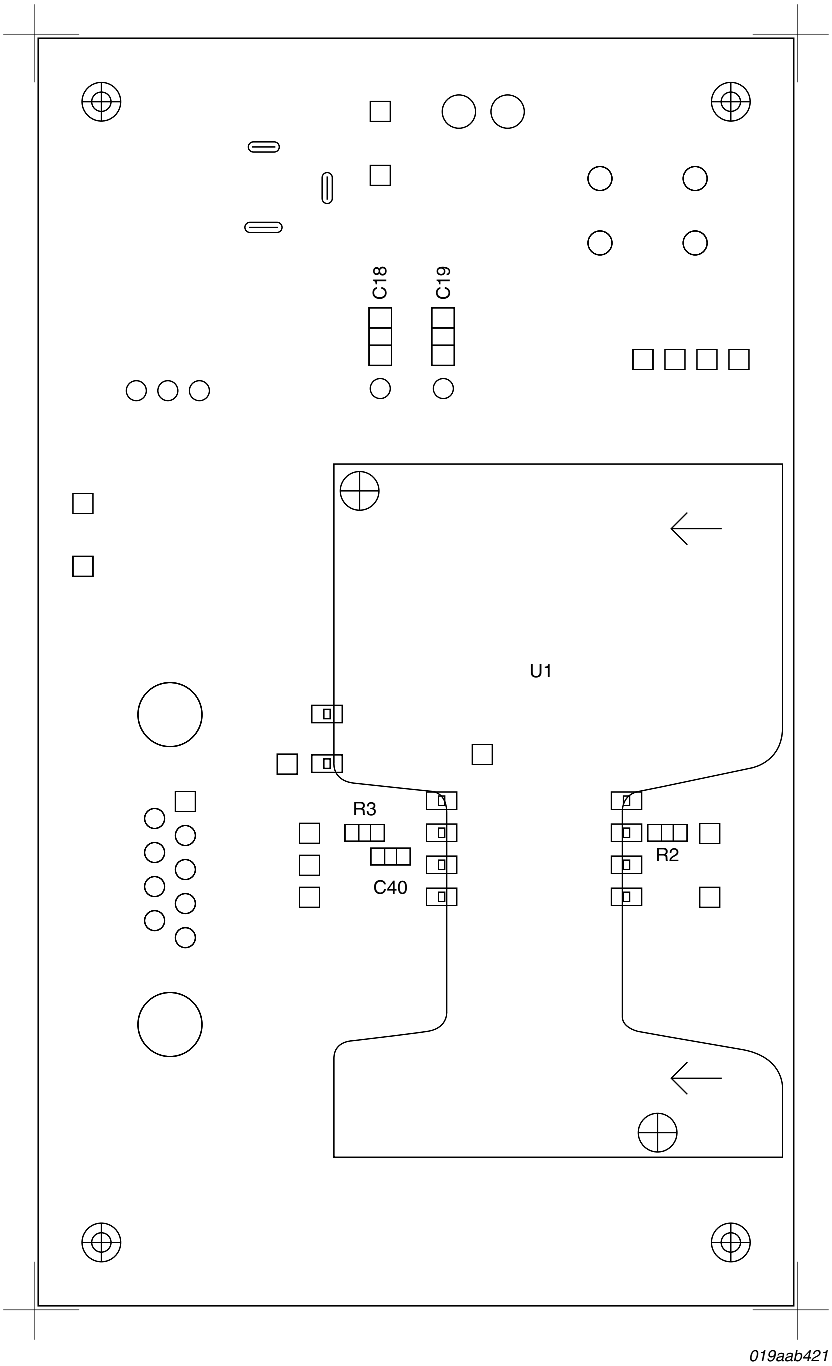
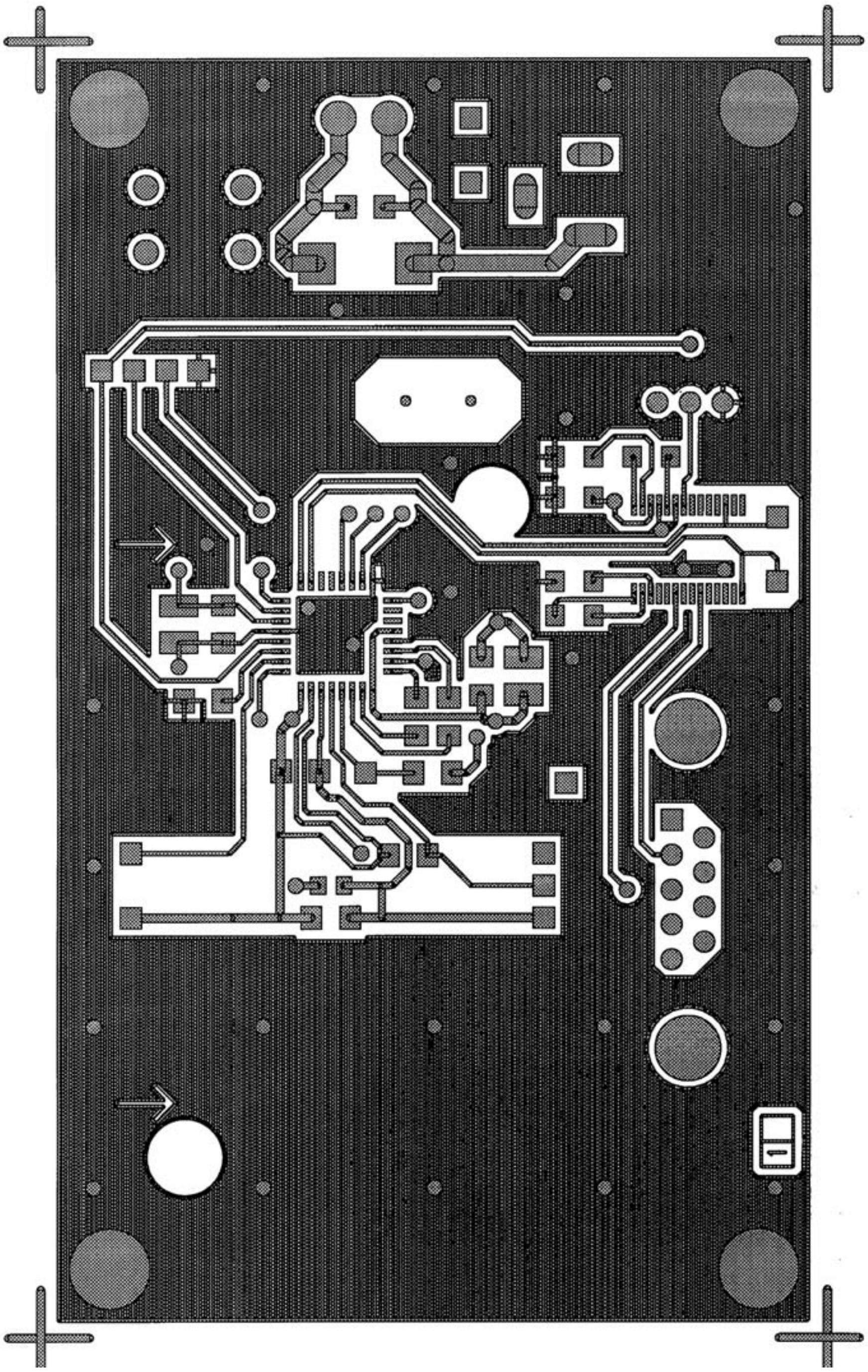


Fig 31. Design example - Components bottom view



019aab422

Fig 32. Design example – Layout top view

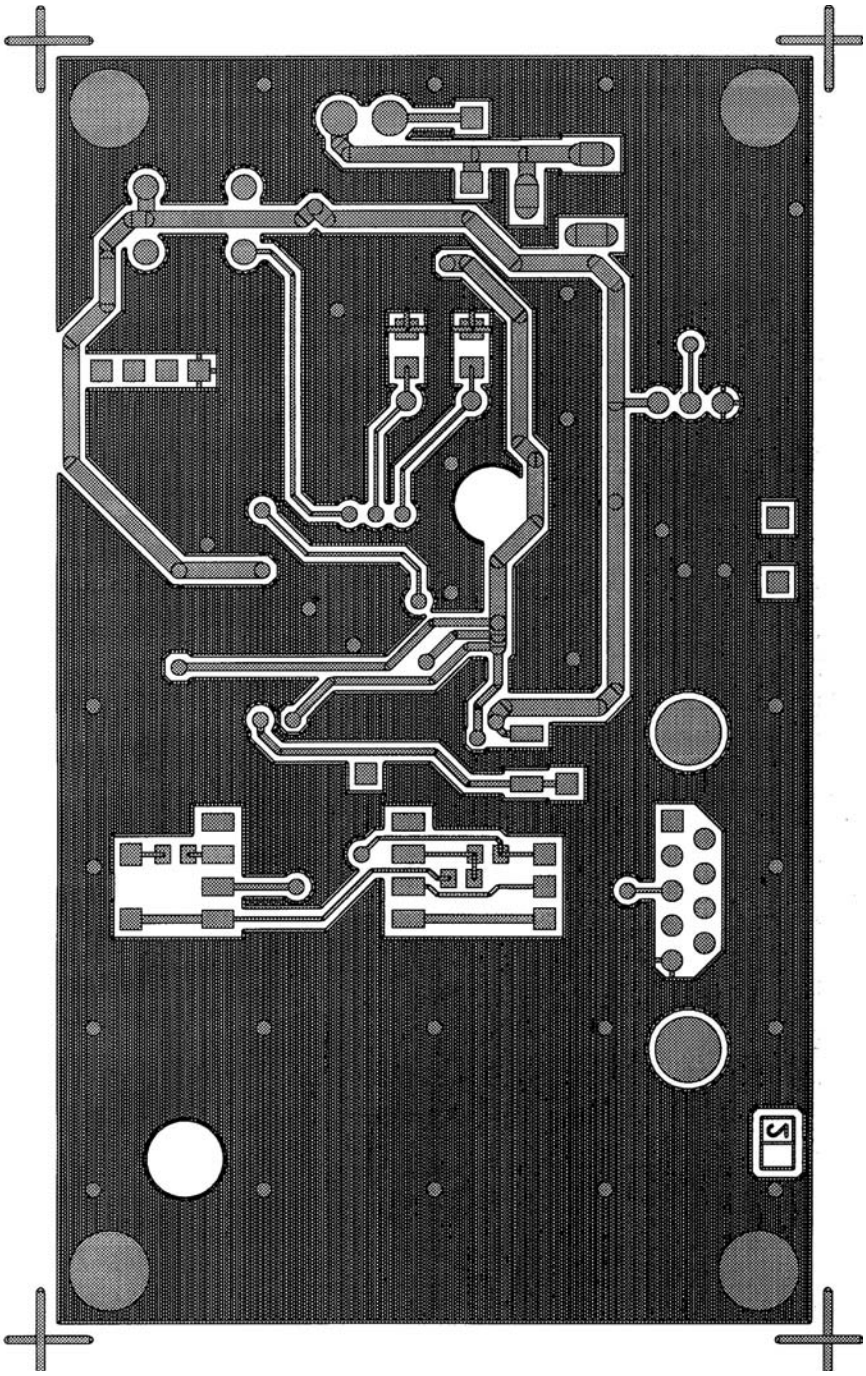


Fig 33. Design example – Layout bottom view

CARTE : PCB793 IND : 4 ETUDE : CAKE8029-11D

PROJET : TDA8029 DATE : 03 Avril 2003

FAIT PAR : ducastel

*** NOMENCLATURE DES COMPOSANTS ***

REFERENCE	GEOMETRY	VALUE	SPECIFICATION : 1/2
BP1	microcosmos	MICROCOSMOS	SECME:MC1033300,Tactile,Switch,Microcosmos
C1	c1206	100nF	Capacitor,CER2,1206,X7R,50V,10%
C2	c293d_d	33uF_16V	Type,293D,Tantal,Capacitor,Package:D,10%
C3	c595d_b	10uF_16V	Type,595D,Tantal,Capacitor,Package:B,10%,T=125C
C4	c1206	100nF	Capacitor,CER2,1206,X7R,50V,10%
C5	c1206	22nF	Capacitor,CER2,1206,X7R,50V,10%
C6	c1206	220nF	Capacitor,CER2,1206,X7R,25V,10%
C7	c1206	100nF	Capacitor,CER2,1206,X7R,50V,10%
C8	c1206	220nF	Capacitor,CER2,1206,X7R,25V,10%
C9	c1206	220nF	Capacitor,CER2,1206,X7R,25V,10%
C10	c1206	220nF	Capacitor,CER2,1206,X7R,25V,10%
C11	c1206	100nF	Capacitor,CER2,1206,X7R,50V,10%
C12	c595d_b	10uF_16V	Type,595D,Tantal,Capacitor,Package:B,10%,T=125C
C13	c293d_a	0.22uF_35V	Type,293D,Tantal,Capacitor,Package:A,10%
C14	c293d_a	0.22uF_35V	Type,293D,Tantal,Capacitor,Package:A,10%
C15	c293d_a	1uF_16V	Type,293D,Tantal,Capacitor,Package:A,10%
C16	c293d_a	1uF_16V	Type,293D,Tantal,Capacitor,Package:A,10%

REFERENCE	GEOMETRY	VALUE	SPECIFICATION : 2/2
C17	c293d_a	1uF_16V	Type,293D,Tantal,Capacitor,Package:A,10%
C18	c1206	22pF	Capacitor,CER2,1206,C0G,50V,5%

019aab424

Fig 34. Design example – BOM 1

C19	c1206	22pF	Capacitor,CER2,1206,C0G,50V,5%
C30	c1206	100pF	Capacitor,CER2,1206,C0G,50V,5%
C40	c1206	1pF	Capacitor,CER2,1206,C0G,50V,5%
IC1	sot358_1	TDA8029	tda8029
IC2	sot341_1	MAX3241CAI	MAXIM:RS232,Tranceivers,Package:SSOP28
J1	mta396d2	MTA396D2	AMP:0-640445-2,Header,Straight,Square,Pin,1x2,Pitch:3.96
J2	jack2.5_h	JACK2.5_H	XXXX:xxxxx,Power,Connector,Horizontal,2.5mm
J3	subd_09fc	D09S13A4GL00	FCI:Delta-D,Connector,Right-Angle,Female,Norm,HE5
K1	int_1k2	090320102	SECME:Switch,2,Positions,Vertical,Short,levers
R1	r0805	0	Resistor,Package:0805,5%,1/8W
R2	r0805	180	Resistor,Package:0805,5%,1/8W
R3	r0805	0	Resistor,Package:0805,5%,1/8W
TP1	tp0.9	TEST_Bar1	Header,Single,Row,Straight,1,pin,h:6to7mm
TP2	tp0.9	TEST_Bar1	Header,Single,Row,Straight,1,pin,h:6to7mm
TP3	tp0.9	TEST_Bar1	Header,Single,Row,Straight,1,pin,h:6to7mm
TP4	tp0.9	TEST_Bar1	Header,Single,Row,Straight,1,pin,h:6to7mm
TP11	tp0.9	TEST_Bar1	Header,Single,Row,Straight,1,pin,h:6to7mm
TP12	tp0.9	TEST_Bar1	Header,Single,Row,Straight,1,pin,h:6to7mm
TP14	tp0.9	TEST_Bar1	Header,Single,Row,Straight,1,pin,h:6to7mm
TP15	tp0.9	TEST_Bar1	Header,Single,Row,Straight,1,pin,h:6to7mm
TP16	tp0.9	TEST_Bar1	Header,Single,Row,Straight,1,pin,h:6to7mm
TP17	tp0.9	TEST_Bar1	Header,Single,Row,Straight,1,pin,h:6to7mm
TP18	tp0.9	TEST_Bar1	Header,Single,Row,Straight,1,pin,h:6to7mm
TP19	tp0.9	TEST_Bar1	Header,Single,Row,Straight,1,pin,h:6to7mm
TP20	tp0.9	TEST_Bar1	Header,Single,Row,Straight,1,pin,h:6to7mm
TP21	tp0.9	TEST_Bar1	Header,Single,Row,Straight,1,pin,h:6to7mm
TP25	tp0.9	TEST_Bar1	Header,Single,Row,Straight,1,pin,h:6to7mm
U1	card_read_ccm01_2251	CCM01_2251	CANON:Card,Read,SMD,8,Contacts
Y1	hc49s	14.745MHZ	KONY:Quartz,Crystal,Low,Profile,Package:HC49S

019aab425

Fig 35. Design example – BOM 2

14. ANNEX III: Part of Implementation Conformance Statement EMV 4.0

Table 16. Implemented protocol types
Please mark the boxes with yes or no

Item number	Protocol type	Reference	Status	Support (Y/N)
1	ATR	*EMV 2000,* §4	m	Y
2	Character protocol T=0	*EMV 2000,* §5.2.2	m	Y
3	Block protocol T=1	*EMV 2000,* §5.2.4	m	Y
4	Transport of APDUs by T=0	*EMV 2000,* §5.3.1	m	Y
5	Transport of APDUs by T=1	*EMV 2000,* §5.3.2	m	Y
6	ATR	ISO 7816-3	o	Y
7	Character protocol T=0	ISO 7816-3	o	Y
8	Block protocol T=1	ISO 7816-3	o	Y
9	Transport of APDUs by T=0	ISO 7816-3	o	Y
10	Transport of APDUs by T=1	ISO 7816-3	o	Y
11	Other protocol	To be precised	o	N

Table 17. General Protocol Information
Please answer the questions by marking the boxes:

Item number	Parameter		Reference	Value(s)
1	Maximum time to issue a warm reset?		EMV 2000, §2.1.3.2	About 41000 CLK
2	Maximum time to issue a deactivation?		EMV 2000, §2.1.5	Terminal dependent
3	Maximum time for the terminal to transmit a command after receiving data from the card?		—	Terminal dependent
			Reference	**Y/N**
4	Does the terminal reject an ICC returning TCK in a T=0 ATR?		EMV 2000, §4.3.4	N
5	Does the terminal continue the card session as soon as all characters indicated in T0 and/or TDi have been received?		EMV 2000, §4.3.4	Y
		Status	**Reference**	**Y/N**
6	Implicit negotiable mode (without PPS)	m	EMV 2000, §4.2, 5	
7	Explicit negotiable mode (with PPS)	o*	EMV 2000, §4.2, 5	

* Outside the scope of the EMV 2000 specification

Table 18. Protocol — Timing

Please answer the questions by marking the boxes with the maximum timing allowed by the terminal before it rejects the ICC behavior

Item number	Parameter	Reference	Status	Maximum allowed*	Maximum supported
1	Maximum ATR duration before warm reset or deactivation?	EMV 2000, §4.4 (4th bullet)	m	20,160 etus	20,160 etus
2	Maximum inter-character time supported before deactivation during ATR?	EMV 2000, §4.4 (3rd bullet)	m	10,080 etus	10,080 etus
3	Default work waiting time in T=0?	EMV 2000, §5.2.2.1	m	D x 10,080 etus	D x 10,080 etus
4	Work waiting time in T=0 with “01” ≤ TC2 ≤ “09”?	EMV 2000, §5.2.2.1	c1	D x (960 x WI etus + 480) etus	D x (960 x WI etus + 480) etus
5	Maximum block waiting time supported before error correction in T=1?	EMV 2000, §5.2.4.2.2	m	(2BWI x 960 x 372 x D / F) + 11+ D x 960 etus	(2BWI x 960 x 372 x D / F) + 11+ D x 960 etus
6	Maximum character waiting time supported before deactivation or block retransmission request in T=1?	EMV 2000, §5.2.4.2.2	m	16 etus for CWI = 1 to 47 etus for CWI = 5	16 etus for CWI = 1 to 47 etus for CWI = 5

c1: applies only if the terminal is able to support such values of TC2.

- The minimum allowed is above the given value in a strict inequality meaning.

Table 19. Parameter Values for ATR

Please supply the supported values

Item number	Param.	Reference	Status	Values Allowed	Values Supported
1	**TS**	*EMV 2000*, §4.3.1	**m**	"3F," '3B"	'3B', '3F'
2	**TA1**	*EMV 2000*, §4.3.3.1	**m**	"11" – "12" and "13"	"11" – "12" and "13"
3	**TA1**	*EMV 2000*, §4.3.3.1	c1	"00" … "10" and "14" … "FF"	(see attached list)
4	**TA1**	*EMV 2000*, §4.3.3.1	**m**	TA1 absent	TA1 absent
5	**TB1**	*EMV 2000*, §4.3.3.2	**m**	"00" (cold reset)	"00" (cold reset)
6	**TB1**	*EMV 2000*, §4.3.3.2	**m**	any value (warm reset)	any value (warm reset)
7	**TB1**	*EMV 2000*, §4.3.3.2	**m**	TB1 absent (warm reset)	TB1 absent (warm reset)
8	**TC1**	*EMV 2000*, §4.3.3.3	**m**	any value *and* (2^{CWI} > (N+1) if T=1)	any value *and* (2^{CWI} > (N+1) if T=1)
9	**TC1**	*EMV 2000*, §4.3.3.3	**m**	TC1 absent	TC1 absent
10	**TD1**	*EMV 2000*, §4.3.3.4	**m**	m.s. nibble: any* l.s. nibble: "0," "1"	m.s. nibble: any* l.s. nibble: "0," "1"
11	**TD1**	*EMV 2000*, §4.3.3.4	**m**	TD1 absent	TD1 absent
12	**TA2**	*EMV 2000*, §4.3.3.5	c1	"00" … '"FF"	'00', '01', '80', '81'
13	**TA2**	*EMV 2000*, §4.3.3.5	**m**	TA2 absent	TA2 absent
14	**TB2**	*EMV 2000*, §4.3.3.6	p**	None	
15	**TC2**	*EMV 2000*, §4.3.3.7	**m**	"0A"	'0A'
16	**TC2**	*EMV 2000*, §4.3.3.7	p**	"00"	
17	**TC2**	*EMV 2000*, §4.3.3.7	C1	"01" … "09" and "0B" … "FF"	"01" … "09" and "0B" … "FF"
18	**TD2**	*EMV 2000*, §4.3.3.8	**m** (T=1)	m.s. nibble: any* l.s. nibble: "1," "E"	m.s. nibble: any* l.s. nibble: "1," "E"
19	**TD2**	*EMV 2000*, §4.3.3.8	p**	l.s. nib. neither "1" nor "E"	
20	**TA3**	*EMV 2000*, §4.3.3.9	**m** (T=1)	"10" … "FE"	"10" … "FE"
21	**TA3**	*EMV 2000*, §4.3.3.9	p**	"00" … "0F," "FF"	
22	**TA3**	*EMV 2000*, §4.3.3.9	**m** (T=1)	TA3 absent	TA3 absent
23	**TB3**	*EMV 2000*, §4.3.3.10	**m** (T=1)	m.s. nibble: "0" … "4" *and* l.s. nibble: "0" … "5" *and* 2^{CWI} > (N+1)	m.s. nibble: "0" … "4" *and* l.s. nibble: "0" … "5" *and* 2^{CWI} > (N+1)

Item	Param.	Reference	Status	Values	
24	**TB3**	*EMV 2000,* §4.3.3.10	p**	m.s. nibble > “4” *or* l.s. nibble > “5” *or* $2^{CWI} \leq (N+1)$	
25	**TB3**	*EMV 2000,* §4.3.3.10	p**	TB3 absent	
26	**TC3**	*EMV 2000,* §4.3.3.11	**m** (T=1)	TC3 = “00”	TC3 = “00”
27	**TC3**	*EMV 2000,* §4.3.3.11	p**	TC3 ≠ “00”	

c1: the terminal may allow specific mode or value only if it is able to support it.

- Provided the values are consistent with characters actually returned.

** Shaded boxes indicate a prohibited capability.

List of supported Fi Di dividers:

0x01, 0x02, 0x03, 0x04, 0x08, 0x11, 0x12, 0x13, 0x14, 0x18, 0x21, 0x22, 0x23, 0x28, 0x31, 0x32, 0x33, 0x34, 0x35, 0x38, 0x41, 0x42, 0x43, 0x44, 0x48, 0x51, 0x52, 0x53, 0x54, 0x55, 0x56, 0x58, 0x61, 0x62, 0x63, 0x64, 0x68, 0x69, 0x91, 0x92, 0x93, 0x94, 0x95, 0x96, 0xA1, 0xA2, 0xA3, 0xA4, 0xA5, 0xA8, 0xB1, 0xB2, 0xB3, 0xB4, 0xB5, 0xB6, 0xC1, 0xC2, 0xC3, 0xC4, 0xC5, 0xC6, 0xC8, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6

Table 20. Protocol T=0 - Parameter Values

Please supply the supported values

Item number	Parameter	Reference	Status	Values	
				Allowed	Supported
1	Terminal minimum inter-character time	EMV 2000, §5.2.2.1 (according to TC1)	m	“12” … “266”	“12” … “266”
2	ICC minimum inter-character time	EMV 2000, §5.2.2.1	m	“11.8”	“11.8”
3	ICC–terminal minimum inter-character time	EMV 2000, §5.2.2.1	m	“15”	“15”
4	Terminal–ICC minimum inter-character time	EMV 2000, §5.2.2.1	m	“16”	“16”

Table 21. Protocol T=1 - Implemented Features

Please mark the boxes with yes or no

Item number	Function	Reference	Status	Support (Y/N)
1	Node addressing: modification of received NAD ≠ “00” during the card session	EMV 2000, §5.2.4.1.1.1	o	Y
2	Byte wise parity checking	EMV 2000, §5.2.5	m	Y
3	Behavior on BWT or WTX excess	EMV 2000, §5.2.5.1	c1-1	N
			c1-2	Y

c1: on BWT or WTX excess, the IFM shall implement one of the two following behaviors:

c1-1: deactivate

c1-2: request for block retransmission

Table 22. Block Types
Please mark the boxes with yes or no.

Item number	Block	Reference	Sending Status	Sending Support (Y/N)	Receipt Status	Receipt Support (Y/N)
1	I-block	*EMV 2000,* §5.2.4.1.1.2	m	Y	m	Y
2	R-block	*EMV 2000,* §5.2.4.1.1.2	m	Y	m	Y
3	Chained I-blocks	*EMV 2000,* §5.2.4.4	m	Y	m	Y
4	S(RESYNCH request)	*EMV 2000,* §5.2.5.1 / 8. and note	c1	Y	m*	Y
4a	S(RESYNCH request)	*EMV 2000,* §5.2.5.1 / 8. and note	c2	N	m*	Y
5	S(RESYNCH response) (c1/c2)	*EMV 2000,* §5.2.5.1 / 8. note	p**		o***	Y
6	S(RESYNCH response) (else)	*EMV 2000,* §5.2.5.1 / 8. note	p**		m***	N
7	S(IFS request) with INF = ‘FE’ is the first block sent following ATR	*EMV 2000,* §5.2.4.3 / 1.	m	Y	na	
8	S(IFS request) otherwise	*EMV 2000,* §5.2.4.3 / 1	p**		m	Y
9	S(IFS response)	*EMV 2000,* §5.2.4.3 / 3.	m	Y	m	Y
10	S(ABORT request)	*EMV 2000,* §5.2.5.1 / 9. and note	p**		m	Y
11	S(ABORT response)	*EMV 2000,* §5.2.5.1 / 9. and note	c3	Y	m*	Y
12	S(WTX request)	*EMV 2000,* §5.2.4.3 / 10.	p**		m	Y
13	S(WTX response)	*EMV 2000,* §5.2.4.3 / 10.	m	Y	m*	Y
14	Vpp error request	*EMV 2000,* §5.2.4.1.1.2 note 8	p**		m*	Y
15	Vpp error response	*EMV 2000,* §5.2.4.1.1.2 note 8	p**		m*	Y

c1: the terminal may issue a S(RESYNCH request) if it supports ISO-compliant resynchronization for proprietary reasons; otherwise, it shall deactivate the ICC contacts when loosing synchronization.

c2: strictly identical to c1 except that the resynch is proprietary and not ISO compliant

c3: when the terminal receives an S(ABORT request), it may issue an S(ABORT response) if it supports abortion; otherwise, it shall deactivate the ICC contacts.

- The ICC sending this type of block is a protocol error; the terminal shall apply *EMV 2000,* §5.2.5 (“Error Detection and Correction for T=1”).

** Shaded boxes indicate a prohibited capability.

*** Optional for cases related to c1/c2; otherwise, mandatory as protocol error.

Table 23. Parameter Values for T=1

Please supply the supported values

Item number	Parameter	Reference	Status	Values	
				Allowed	Supported
1	LEN of INF	EMV 2000, §5.2.4.1.1.3 reference specification	m	“0” … “254”	“0” … “254”
2	IFSD	EMV 2000, §5.2.4.2.1	m	“254”	“254”
3	TC1 minimum	EMV 2000, §5.2.4.2.2	m	“11” … “266”	“11” … “266”
4	ICC intercharacter minimum time	EMV 2000, §5.2.4.2.2	m	“10.8”	“10.8”
5	ICC Block guard time	EMV 2000, §5.2.4.2.2	m	“21”	“21”

15. ANNEX IV: Specific Case 4 command correction with Mask 07

15.1 ANALYSE OF THE ISSUE

This issue occurs when the following conditions are fulfilled:

Protocol T= 0,

C-APDU of case 4 type i.e. CLA INS P1 P2 Lc [Lc Data] Le when Le is equal 1 or 2,

The warning status words 62 h XXh is returned by the card,

The number of returned data (excluding the status words) is equal to 1or 2.

In EMV4.0 document, this transaction is named ‘Case 4 command with warning condition’.

15.2 Correct transaction with TDA8029C207

HOST	**READER**	**CARD**
CLA INS P1 P2 Lc [Lc Data] Le =>		
	CLA INS P1 P2 Lc =>	
		<= INS
	[Lc Data] =>	
		<= 62 XX
	00 C0 00 00 00 =>	
		<= 6C YY
	00 C0 00 00 YY =>	
		<= C0 [YY Data] 90 00

The R-APDU [YY Data] 62 XX is returned to the host.

15.3 Transaction performed by the TDA8029C206

HOST	**READER**	**CARD**
CLA INS P1 P2 Lc [Lc Data] Le =>		
	CLA INS P1 P2 Lc =>	
		<= INS
	[Lc Data] =>	
		<= 62 XX
	00 C0 00 00 00 =>	
		<= 6C YY
	00 C0 00 00 YY =>	
		<= C0 [YY Data] 90 00

Instead of returning [YY Data] 62 XX, the TDA8029C206 sends back the R-APDU [YY Data] 90 00 when YY is equal 1 or 2.

16. Annex V: Recommendation rules when using the I2C interface

First, the IIC clock shall not exceed 60 KHz and needs to have a 50 % duty cycle.

You should insert 6 µs delay between the start condition and clock low and 15 µs delay before and after the ACK.

The IIC master shall support the clock stretching mechanism in order to work properly with the TDA8029.

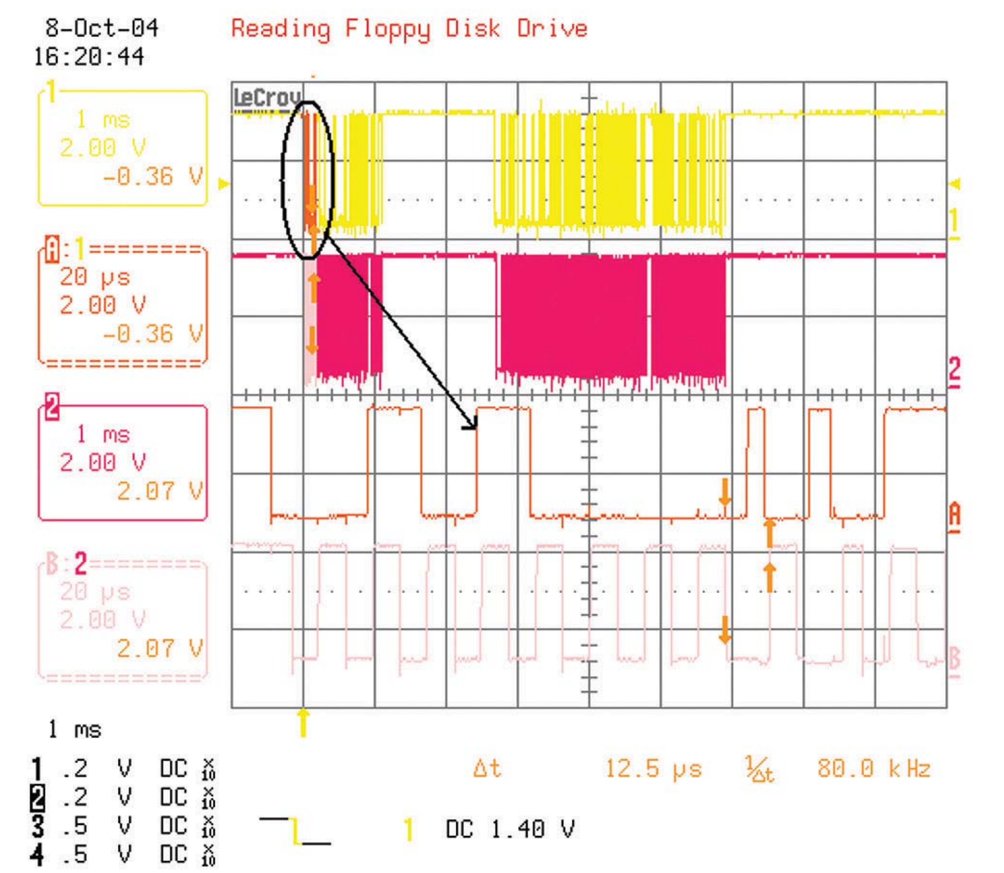
For any exchange, a write command followed by a read command need to be sent in order to complete the exchange.

A delay may be inserted between the write command and the read command otherwise the TDA8029 may not acknowledge the address saying it is not yet ready (see picture 3).

STA 50 60 00 00 0A 6A **STO** (Write command for mask number)

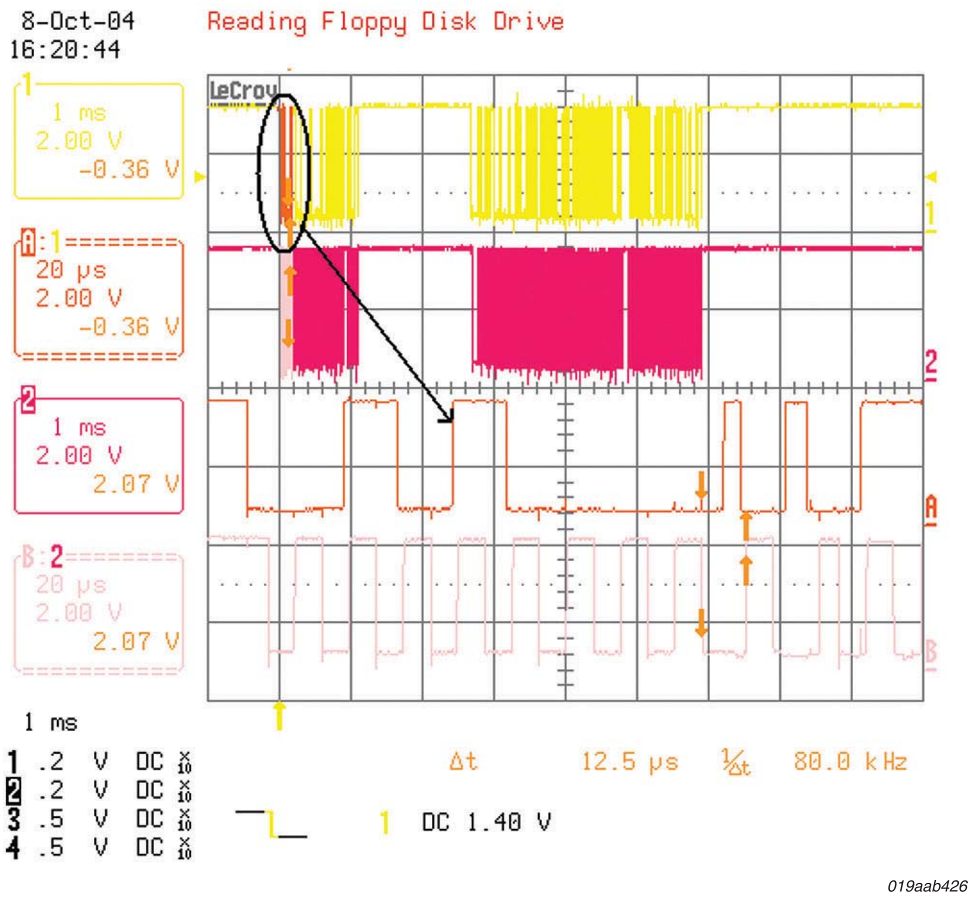
STA 51 60 00 0E 0A 30 37 20 52 65 6C 65 61 73 65 20 31 2E 30 05 **STO** (Read command for Mask7 release 1.0)

You will find bellow some scope pictures showing a “Mask Number” command exchanged between the PC and the TDA8029 demo board (60Kbauds).



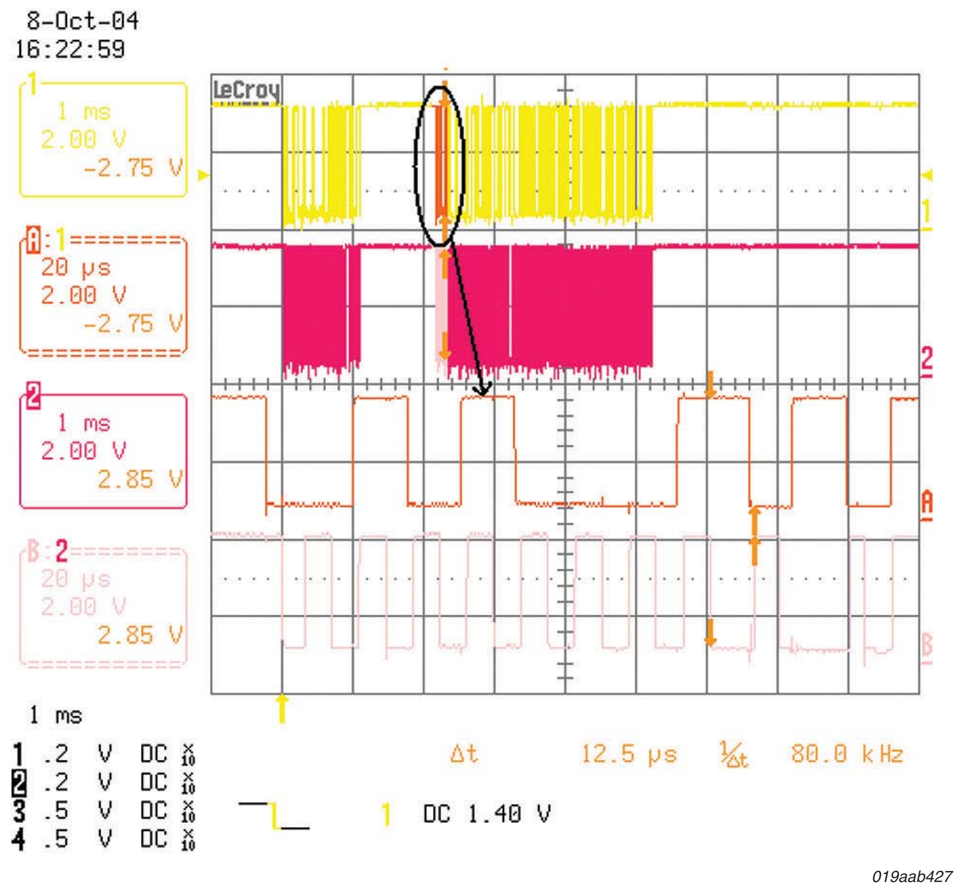
(1) Channel 1: SDA; Channel2: SCL

Fig 36. Zoom of the Mask number command from Host



(1) Channel 1: SDA; Channel2: SCL

Fig 37. Zoom of the answer from TDA8029



(1) Channel 1: SDA; Channel2: SCL

Fig 38. Zoom of the NAK of the address during the Answer from TDA8029

17. Legal information

17.1 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

17.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from national authorities.

Evaluation products — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer.

In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages.

Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

17.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are property of their respective owners.

18. List of figures

Fig 1. TDA8029 application diagram example 3
Fig 2. ESM Mechanism...5
Fig 3. CDEL pin action overview................................6
Fig 4. Correspondence between reset delay and CDEL capacitor...7
Fig 5. Frame structure...8
Fig 6. 4 header bytes 8
Fig 7. Successful command frame description...........9
Fig 8. Unsuccessful command frames.......................9
Fig 9. Acknowledge frame 9
Fig 10. RS232 Interface configuration........................10
Fig 11. Communication initiated by the system controller 11
Fig 12. Communication initiated by the system controller 11
Fig 13. Communication initiated by the TDA8029.......12
Fig 14. Communication initiated by the TDA8029 – Frame description 12
Fig 15. Card move – frame example..........................13
Fig 16. I2C Bus interface configuration......................15
Fig 17. I2C communication with ESM off 16
Fig 18. I2C communication with ESM on 17
Fig 19. 4 lines I2C communication example initiated by the host 18
Fig 20. 4 lines I2C communication example initiated by the TDA8029..18
Fig 21. 2 lines I2C communication.............................19
Fig 22. I2C write protocol 20
Fig 23. I2C read protocol 20
Fig 24. I2C read protocol - Error 21
Fig 25. power_up_5V frame exchange 31
Fig 26. Read IO command – VAL byte description 42
Fig 27. SCRTester main window................................46
Fig 28. SCRTester – Serial port connection error window..47
Fig 29. Design example - Electrical schematic 48
Fig 30. Design example - Components top view.........49
Fig 31. Design example - Components bottom view...50
Fig 32. Design example – Layout top view 51
Fig 33. Design example – Layout bottom view 52
Fig 34. Design example – BOM 1 53
Fig 35. Design example – BOM 2 54
Fig 36. Zoom of the Mask number command from Host 63
Fig 37. Zoom of the answer from TDA8029................64
Fig 38. Zoom of the NAK of the address during the Answer from TDA8029...................................65

19. List of tables

Table 1. Host interface hardware configuration 4
Table 2. ESM hardware configuration............................5
Table 3. I2C modes...15
Table 7. List of error codes 24
Table 8. Error codes for each command.......................26
Table 9. Baud rate parameter 29
Table 10. set_clock_card parameter...............................34
Table 12. IO coding in set_IO command........................42
Table 13. VAL coding in set_IO command.....................42
Table 14. Host interface hardware configuration 45
Table 15. ESM hardware configuration..........................45
Table 16. Implemented protocol types...........................55
Table 17. General Protocol Information 55

20. Contents

1. Introduction 3
2. Hardware and power management 4
2.1 Hardware 4
2.2 Host Controller Interface 4
2.3 Power management: Energy Saving Mode 4
2.4 Shutdown mode 6
2.5 Power-On Reset 6
3. Software aspect 7
4. Protocol “ALPAR” 8
4.1 General dialog structure 8
4.1.1 Successful command 9
4.1.2 Unsuccessful command 9
4.1.3 Answer with an acknowledge (power_off, idle_mode, power_down_mode) 9
5. RS232 Interface 10
5.1 General description 10
5.2 Dialog structure in case of Energy Saving Mode activated 10
5.2.1 Communication initiated by the system controller 11
5.2.2 Communication initiated by the TDA8029 12
5.2.3 Timing considerations 13
5.2.4 Abnormal communication process 14
6. I2C-bus interface 15
6.1 General description 15
6.2 Energy Saving Mode deactivated 16
6.3 Energy Saving Mode activated 17
6.3.1 4 lines I2C-bus 17
6.3.2 2 lines I2C-bus 19
6.4 Data link layer 19
6.5 I2C transactions 20
6.5.1 I2C write command message 20
6.5.2 I2C read command message, normal answer 20
6.5.3 I2C read message, error answer 21
6.5.4 Examples of I2C transactions 21
7. Command bytes 22
7.1 General commands 22
8. Error list 24
9. Commands description 28
9.1 General commands 28
9.1.1 send_num_mask 28
9.1.2 check_card_presence 28
9.1.3 get_reader_status 28
9.1.4 set_serial_baud_rate 29
9.1.5 set_esm_properties 29
9.1.6 time_out 30
9.1.7 idle_mode (clock stop low) 30
idle_mode (clock stop high) 30
9.1.8 power_down_mode 30
9.2 Asynchronous card related commands 31
9.2.1 power_up commands 31
9.2.1.1 power_up_5V 31
9.2.1.2 power_up_3V 32
9.2.1.3 power_up_1.8V 32
9.2.1.4 power_up_iso 32
9.2.2 power_off 32
9.2.3 card_command (APDU) 32
9.2.4 process_T=1_command 33
9.2.5 negotiate 33
9.2.6 Ifsd_request 34
9.2.7 set_clock_card 34
9.2.8 card_take_off and card_insertion 34
9.2.9 set_card_baud_rate 35
9.2.10 set_nad 36
9.2.11 get_card_param 36
9.3 Synchronous card related commands 37
9.3.1 Synchronous card S=9 37
9.3.1.1 power_up_S9 37
9.3.1.2 read_S9 37
9.3.1.3 read_S9_protection 37
9.3.1.4 write_S9_protected 38
9.3.1.5 write_S9_unprotected 38
9.3.1.6 verify_pin_code 38
9.3.1.7 compare 38
9.3.2 Card S=10 39
9.3.2.1 power_up_S10 39
9.3.2.2 process_S10 39
9.3.3 I2C cards 40
9.3.3.1 power_up_I2C 40
9.3.3.2 read_I2C 40
9.3.3.3 read_I2C_extended 40
9.3.3.4 read_current_I2C 41
9.3.3.5 write_I2C 41
9.4 General purpose IO commands 42
9.4.1 read_IO 42
9.4.2 set_IO 42
10. Information field for asynchronous cards 43
10.1 Extended cases 44
11. Conclusion 45
12. ANNEX I: SCRTESTER 46
12.1 Installation 46

continued >>

12.2 Run SCRTester 46
12.2.1 The top right window 47
12.2.2 The bottom right window 47
12.2.3 The left window: card parameters 47
13. ANNEX II: Hardware information.................... 48
14. ANNEX III: Part of Implementation Conformance Statement EMV 4.0 55
15. ANNEX IV: Specific Case 4 command correction with Mask 07.................................. 62
15.1 ANALYSE OF THE ISSUE.............................. 62
15.2 Correct transaction with TDA8029C207 62
15.3 Transaction performed by the TDA8029C206 .. 62
16. Annex V: Recommendation rules when using the I2C interface 63
17. Legal information 66
17.1 Definitions 66
17.2 Disclaimers... 66
17.3 Trademarks 66
18. List of figures.. 67
19. List of tables 68
20. Contents.. 69

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 2 February 2011

Document identifier: AN10207